Towards Dependable Home Networking: An Experience Report

Yi-Min Wang
Wilf Russell
Anish Arora
Jun Xu
Rajesh K. Jagannathan

April 18, 2000

Technical Report
MSR-TR-2000-26

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

To appear in *Proc. IEEE International Conference on Dependable Systems and Networks (formerly FTCS)*, June 2000.

Towards Dependable Home Networking: An Experience Report

Yi-Min Wang *Microsoft Research Redmond, WA*
Wilf Russell *Microsoft Research Redmond, WA*
Anish Arora *Ohio State Univ. Columbus, OH*
Jun Xu *Univ. of Illinois Urbana, IL*
Rajesh K. Jagannathan *Ohio State Univ. Columbus, OH*

Abstract

As the success of the Web increasingly brings us towards a fully connected world, home networking systems that connect and manage home appliances become the natural next step to complete the connectivity. Although there has been fast-growing interest in the design of smart appliances and environments, there has been little study on the dependability issues, which is essential to making home networking part of our daily lives. The heterogeneity of various in-home networks, the undependable nature of consumer devices, and the lack of knowledgeable system administrators in the home environment introduce both opportunities and challenges for dependability research. In this paper, we report the dependability problems we encountered and the solutions we adopted in the deployment of the Aladdin home networking system. We propose the use of a soft-state store as a shared heartbeat infrastructure for monitoring the health of diverse hardware and software entities. We also describe a system architecture for connecting powerline devices to enhance dependability, and a monitoring tool for detecting unusual powerline activities potentially generated by intruders, interferences, or ill-behaved devices.

1. Introduction

With the explosive growth of the Web, we are increasingly moving towards a fully connected world. As broadband communication is being brought to homes with accelerating speed and as small handheld devices get smarter, more popular, and better connected, the notion of being able to communicate with anything at any time from anywhere is bound to become a reality. In this big picture, home networking is a natural next step in which both existing devices and future smart appliances are fully connected inside the house and accessible to the homeowners whenever needed. Starting from the simple scenarios of sharing files, printers, and Internet connections, home networking is also moving towards enabling multi-player, multi-PC games, digital video and audio anywhere in the house, device automation, remote diagnosis of home appliances, etc. An informal survey shows that different people have dramatically different ideas on what the killer applications for home networking should be. It is therefore important to provide an infrastructure for robust device connectivity to allow the construction of versatile applications on top of the infrastructure.

In the *Aladdin* research project [WRA00], we focus on providing the system infrastructure for device connectivity by integrating the seven in-home networks into one dependable home network: powerline, phoneline, RF (Radio Frequency), IR (InfraRed), A/V LAN, security, and temperature control. The goal is to allow the users to plug in a device on any of these networks and make it part of the Aladdin system so that it can be used in conjunction with all the other devices to accomplish higher-level system- or user-directed tasks. To make the whole system good enough to live with, one must pay special attention to the dependability issues, including reliability, availability, security, and manageability. The second goal of the Aladdin project is to support dependable remote home automation and sensing. We believe that home networking adds significant value even when people are away from their homes. Therefore, providing reliable and secure remote access to home networks and providing reliable sensing and control of devices are important parts of the project.

Home networking introduces several new challenges in the area of dependability. First, in the consumer electronics market, selling large volumes in order to drive the price down is a key to success. Manufacturers are therefore led to packaging their products as add-on modules with primitive I/O specifications so that they can be used with a variety of different systems and can be added incrementally to existing systems to control new or existing devices. However, such a design creates dependability problems that must be dealt with by the system. The problems with the powerline-based modules and sensors, which will be described later, are good examples. Second, home networks are heterogeneous and dynamic. Each of the in-home networks has a different characteristic in terms of bandwidth, connectivity, security, interferences, etc. This provides a new opportunity to exploit the redundancy provided by one network to solve dependability problems faced by another network. In addition, compared to the machines in the enterprise environment, consumer devices in the home networking environment are more dynamic in terms of mobility, availability, and extensibility. The system must be able to keep track of all the changes in the entire network in order to support reliable operation. Finally, enterprise environments usually rely on human administrators to

perform tasks such as failure diagnosis and recovery, and intrusion detection and defense. But, in the home networking domain, we cannot afford to have the same level of administrator support and so the system must automatically perform those tasks as much as possible. What makes it even more challenging is the fact that consumer devices fail more often and with more different modes, and intrusions and interferences can come from different networks. Building a dependability framework that monitors, diagnoses, and recovers from known dependability problems and allows for extensibility to accommodate new failure modes as they are observed is the centerpiece of the Aladdin project.

In this paper, we focus on three dependability problems and their solutions. First, to robustly track the health of the diverse network entities including devices, sensors, objects, daemon processes, etc., we propose the use of a soft-state store as a shared heartbeat infrastructure. Second, to enhance the dependability of powerline control operation, we describe a system architecture that makes use of private powerline networks and the phoneline network. Finally, to automatically detect and diagnose unreliable device behaviors and even security intrusions, we provide a general tool for monitoring powerline activity.

2. Overview of the Aladdin System

In an ideal home networking system, the house is wired for running Ethernet and most devices are smart, networked devices connected directly to the Ethernet and running device control software themselves. A *home gateway* machine sits between the home network and the external communication infrastructures including the Internet and telephony. *User Access Points (UAPs)* are wall-mounted or stand-alone flat-panel displays deployed throughout the house to allow convenient access to in-home information (calendars, etc.) as well as the Internet from anywhere in the house. UAPs also expose Web-based, natural language-based, and voice-based interfaces for remotely controlling household devices and for monitoring environmental factors through remote sensors. Network bridges are provided for bridging devices on other communication media such as the powerline, RF, IR, and A/V cables to the Ethernet backbone.

Since smart devices are not yet generally available, the current Aladdin system accommodates existing devices by using six Windows 98 PCs and their peripherals to serve as both User Access Points and network bridges. The PCs are all connected by 1Mb/s or 10Mb/s Ethernet over the phoneline [H98]. They also act as device proxies by running device control software on behalf of the devices. The system, currently consisting of about 60 devices, is deployed in the first author's three-story house and used by the author's family on a daily basis.

The software architecture of the Aladdin system is divided into three layers. At the bottom, the *system infrastructure layer* consists of the soft-state store, its associated publish/subscribe eventing component, an Attribute-Based Lookup Service for maintaining a database of all available devices, a Name-Based Lookup Service for maintaining a table of all running object instances, and a device announcement protocol for bridging non-Ethernet devices. These components will be discussed in more details in Section 3. The infrastructure layer also contains the system management daemons for PC failure detection and recovery, which are omitted in this paper due to space limitation.

In the middle, the *application layer* consists of device objects and device daemons that encapsulate device- and network-specific details, and home networking applications for control and sensing. At the top, the *user interface layer* provides a browser-based point-and-click interface, a text-based natural language interface, and a voice recognition interface. To enable remote home automation, Aladdin allows email-based control using natural language and provides emergency notifications through the text messaging support of cell phones. For example, in the current deployment, the homeowner can send a digitally signed and encrypted email to close the garage door, request short video clips from the surveillance cameras, or turn on and off electrical appliances, etc. He will also receive a cell phone call when the garage door is opened, the fish tanks are leaking water, or a power outage occurs, etc.

3. Soft-State Store as a Shared Heartbeat Infrastructure

Home networking systems are more heterogeneous and dynamic than other typical distributed systems. Diverse devices and sensors connected to various in-home networks, and transient objects and long-lived daemon processes running different protocols contribute to the heterogeneity. The consumer-market nature of these network entities implies that they will fail, move, and disconnect more often than their enterprise counterparts, making the system more dynamic. An essential task of any home networking system is therefore to keep track of the status of these network entities. In Aladdin, we use two lookup services to accomplish this task: the *Attribute-Based Lookup Service (ABLS)* maintains a database of available devices and sensors, and supports queries based on device attributes including device type, physical location, etc.; the *Name-Based Lookup Services (NBLS)* maintains a table of running object instances, and provides the name-to-addresses mapping. An object can have multiple addresses including, for example, a distributed object interface pointer in its marshaled, string form and a queued address for asynchronous communication.

To keep track of the health and availability of the network entities, either the system needs to ping them or they need to send periodic heartbeats to the system. The latter approach is the preferred one in home networking for several reasons. First, to reduce cost, many consumer sensors are transmitters only and do not support polling. Also, some of the sensors already periodically send refreshes of their states, which effectively provide heartbeat information. Second, many network protocols and programming paradigms (distributed objects, messaging, etc.) are likely to coexist in home networking systems due to market competition. It will not be practical to require the system to be able to ping all devices and objects with various existing and future protocols and paradigms. Moreover, the ping interface provided by some objects may hang, compromising system robustness. Finally, the pinging approach would require the system to persist information regarding the existence and ping interfaces of all network entities in order to handle system failures. In contrast, the heartbeat approach can simply rely on the refreshes to reconstruct lookup service entries.

Since supporting heartbeats is at the core of building a robust home networking system, we took the approach of building a *Soft-State Store (SSS)* as a shared heartbeat infrastructure and layering lookup services on top of the store. The term "*soft-states*" is defined as *volatile or nonvolatile states that will expire if not refreshed within a pre-determined, but configurable amount of time* [WRA00]. (The notion of soft-states is similar to that of leasing [E99].) The SSS APIs allow programs to create soft-state types and variables, specify heartbeat intervals and maximum number of missing heartbeats, update and retrieve soft-state variable values, and subscribe to events related to changes in the store. The SSS daemon running on each machine is responsible for replicating changes to other machines and for firing events to local subscribers.

The Aladdin system management daemons use the SSS directly to detect the failures of machines and critical daemon processes. Device control objects send heartbeats to the NBLS, which relies on the SSS to maintain the lookup service entries. Sensors emit periodic state refreshes that are received by device daemons and translated into refreshes of corresponding ABLS entries. The ABLS stores the entries in a database to support rich queries, but it relies on the SSS to time out expired entries and fire appropriate events. For example, when a battery-operated garage door sensor runs out of battery, its ABLS entry will eventually be timed out so that the garage door opener object will not utilize incorrect, stale sensor state and an alert can be sent to the homeowner to remind him/her to replace the battery. The interactions between the devices and the SSS are more involved and will be described in more details next.

Aladdin Device Adapter

The most popular off-the-shelf home automation modules today are add-on powerline modules that sit between the electric outlets and the ordinary devices to be controlled remotely. There are two dependability problems in the use of these add-on modules. First, the On/Off status of an add-on module may not be consistent with that of the device that plugs into it. When the device is broken, physically switched off, or unplugged, it is no longer controllable by the system but the add-on module can still be turned on and off. Second, when both the module and the device are unplugged from the outlet, the device may still be incorrectly listed in the lookup service as available.

To achieve dependable operations with add-on modules and to eliminate the need of manual de-registration and re-registration every time a device is physically moved, we introduce the concept of an *Aladdin Device Adapter* as an enhanced add-on module. The key idea is that the Adapter periodically announces the device type, physical location, etc. of the device on its behalf. The Adapter is also responsible for detecting that the device is no longer controllable, and notifying the system to remove its lookup service entry. When both the Adapter and the device are disconnected from the outlet, the missing heartbeats from the Adapter will allow the system to eventually time out its entry.

To demonstrate the Adapter concept, we built a prototype for use with the popular X10 powerline control protocol [S98]. The Adapter consists of (1) an AC current detector that monitors the real working status of the attaching device by measuring the AC current flowing through the device; (2) a regular X10 receiver module that responds to remote On/Off commands by gating the AC current supplied to the device; (3) a state machine that, based on the status of the current detector and the receiver module, decides when to perform device joining and leaving announcements. An initial investigation of the hardware and software requirements suggested that off-the-shelf consumer modules could be modified into the Adapters with minimum additional circuitry.

Initially, when a device (for example, a lamp) is plugged into the Adapter and the Adapter is plugged into a wall outlet, both the power switch on the device and the X10 module in the Adapter are in the Off position. The X10 address of the Adapter is set to that assigned to the outlet. (Every interesting outlet in the house has been assigned a unique X10 address, which maps to a unique physical location, for example, "the garage side of the kitchen on the first floor".) By using the manual override function provided by X10, a user can turn on both the device and the module by simply turning on the power switch on the device. Upon detecting the state change, the state machine sends out a device-joining announcement over the powerline in the form of an *extended X10 code*.

The code contains (1) the X10 address of the outlet that identifies both the powerline address for controlling the device and the physical location of the device; (2) a *device code* that is pre-assigned to represent a particular type of devices; and (3) a *module code* that specifies the valid commands that can be sent to the Adapter to control the device. Upon receiving the announcement, a PC decodes the above information out of the extended X10 code, and registers the device with the ABLS. Afterwards, the Adapter periodically sends out the same announcement as long as the device is still controllable. The PC then performs periodic refreshes of the ABLS entry to prevent it from being timed out by the soft-state store.

When the power switch on the device is turned off or when the device is broken, the state machine detects that the X10 module is still on but the AC current detector does not detect any current flowing through the device. It concludes that the device is no longer controllable by the Aladdin system and so sends out a device-leaving announcement on behalf of the device, again in the form of an extended X10 code over the powerline. Upon receiving the announcement, a PC notifies the ABLS of the device's unavailability to preserve the consistency of the lookup service.

4. System Architecture for Enhancing Powerline Control Dependability

Powerline networking is likely to be an essential part of any home networking systems because it provides the most ubiquitous wired connectivity throughout the majority of the houses. However, due to the quality of the physical wiring and the inherently less secure connection topology [Ev96], powerline networking suffers from more dependability issues than phoneline networking. As a result, the powerline networking industry has advanced less than the phoneline networking industry in terms of both supported bandwidth and programming abstraction.

In the current Aladdin prototype, we use the X10 powerline control protocol and devices because they are the most consumer-ready products at this point, and build system-level solutions to achieve dependability. It remains to be seen whether the next-generation powerline networking protocols can succeed in the consumer market by solving most of the dependability problems at the network level with reasonably low costs in order to compete with other alternatives such as phoneline and wireless networking. With the understanding that X10 has a few inherent weaknesses that cannot be masked by the system, we expect their uses in controlling more critical home appliances will most likely be replaced in the future. In this paper, we focus on the more generic powerline dependability issues and omit discussions on X10-specific issues such as command atomicity and signal collisions.

Security is probably the No. 1 concern. Most houses share the same "powerline subnet" with some neighboring houses connected to the same distribution transformer. Powerline commands from one house can potentially reach the devices in another near-by house and interfere with the controlling of those devices. Conversely, powerline commands and device announcements from one house can potentially be monitored by another house, thus creating privacy concerns. A canonical solution to this problem is to rely on digital signatures and encryptions. But the limited bandwidth currently achievable by low-cost consumer products poses a challenge to the applicability of this solution.

Reliability is also a big issue in powerline networking. Powerline control modules are delicate electrical components and, since they are directly plugged into wall outlets, they are susceptible to the damage by voltage spikes. Signal attenuation may prevent powerline commands generated by a controller connected to one circuit breaker from reliably reaching the target device connected to another circuit breaker. Line noises generated by some household appliances or external sources may transiently interfere with the operation of powerline controls. Finally, since the most common usage of powerline networking is to enable wireless remote control by the users, one or more RF transceivers are almost always present. Unfortunately, such transceivers also provide channels for RF interferences to create either transient, intermittent, or persistent reliability problems.

Figure 1 illustrates the system architecture used in our current deployment for addressing the above dependability issues. We installed an X10 signal blocker at the main electrical panel to block X10 signals from coming into and leaving the house. For critical devices, we rely on the more secure phoneline to provide additional security. The basic idea is to use the phoneline to reach *private powerline networks* to which critical devices are connected. The private powerline networks are constructed by using an X10 signal filter to isolate a power strip from the common powerline (see Figure 1). For example, the garage door opener in Aladdin resides on a private powerline network that is configured as a peripheral of the garage PC. To remotely control the garage door opener, one must go through the phoneline Ethernet to reach the garage PC and send out X10 commands from there. Even as the next-generation powerline networking protocols provide better and better security, the concept of exploiting multiple redundant networks to provide additional security will remain valuable.

On the reliability side, we installed a whole-house surge protector at the main electrical panel to absorb potential power surges. To address the issue of signal attenuation, we again exploit the multiple redundant networks. Suppose PC "Abu" needs to control an Adapter but cannot reach it

directly over the powerline. As part of the ABLS lookup operation for locating devices, "Abu" would have identified the subset of other PCs that have been able to receive the announcements from the target Adapter and therefore can directly control the Adapter. "Abu" then issues its control command by first routing the command over the phoneline (in the form of a distributed object method call) to one of those PCs and then sending out a powerline command from there.

Powerline-based motion sensors introduce potential reliability problems. Since they are designed to quickly detect motions and fire events by sending powerline commands, having multiple motion sensors in the presence of persistent motions can exhaust the already limited bandwidth. Our solution is to place each of them on a separate private powerline network as shown in Figure 1, and let each PC be responsible for recording its local motion sensor state. These sensor states are then propagated to other nodes over the phoneline.

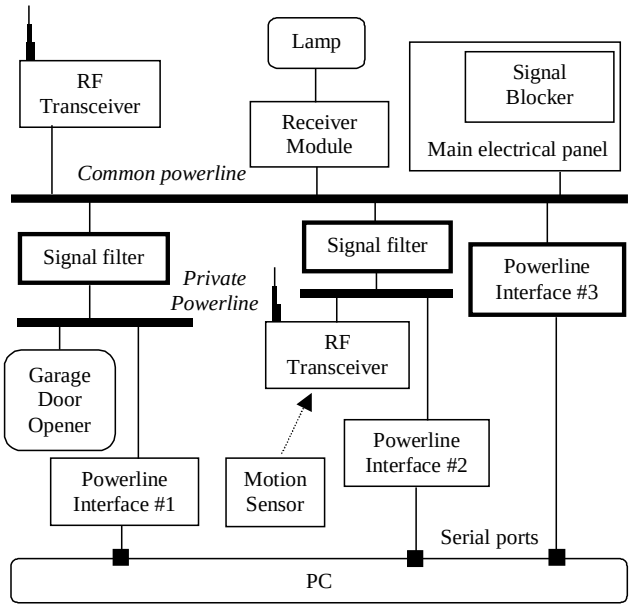

Figure 1. System architecture for enhancing powerline control dependability.

5. Powerline Activity Monitoring

We have also observed reliability problems associated with powerline transmitters. For example, during a 24-hour period, one of the RF transceivers kept receiving RF interferences that resembled valid X10 wireless signals. As a result, it kept converting those interferences to powerline signals and consumed all the bandwidth. In two other incidents, a faulty powerline interface kept generating random X10 signals, again saturating the powerline.

The above observations suggest that consumer device-control transmitters may exhibit non-fail-stop behaviors and create reliability or even security problems if some of the random commands they generate happen to address critical devices. These problems are likely to remain even when the next-generation powerline networking protocols provide better security. The private powerline networking solution mentioned previously can help alleviate this type of problem. But a general tool for monitoring powerline activity is needed to automatically detect and diagnose these problems. We describe such a tool in this subsection. Although the examples are X10-specific, the concept is applicable more generally.

To deal with the RF transceiver interference and faulty powerline interface problems, Aladdin uses *pattern-based detection* [K94]: it monitors X10 powerline transmissions and detects whether they satisfy some specified patterns of bad behaviors. In the X10 system, each receiver module is manually assigned an X10 address consisting of a *house code* (A through P) and a *unit code* (1 through 16). A typical X10 transmission sequence consists of some number of *address commands* (e.g. "A2", "A5", etc.) of the same house code, followed by one or more *function commands* of that same house code (e.g., "A On", "A Off", etc.). Since each transceiver is tuned to receive and transmit X10 commands of exactly one house code, the bad pattern we observed in the RF interference scenario is relatively simple to model as follows.

> *"Within the last ΔS time, K or more identical function commands were transmitted consecutively on the powerline"*

For the faulty interface problem, the pattern of bad behavior is more complex:

> *"Within the last ΔT time, there were L or more transmission bursts on the powerline in each of which either M or more address commands with the same unit code were transmitted consecutively or N or more identical function commands were transmitted consecutively"*

In these two expressions, K, L, M, and N are integer constants (roughly, based on our incidence logs, 25, 15, 5, and 5) and ΔS and ΔT are time constants (roughly, 2 minutes each).

We have chosen not to implement monitors for each observed pattern on a case-by-case basis, since we expect the set of bad behavior patterns to grow as we collect new incidence reports and gain further experience with the fault types in consumer devices. Building upon the SIEFAST monitoring language and its implementation [S], our approach is to express all observed patterns in an extended regular expression language and to automatically generate the monitoring daemon. The extended regular expression language allows (1) parameterized matching of events, for

succinct specification of bad behaviors, and (2) event matching in conjunction with satisfaction of state predicates, i.e. Boolean expressions on the system state, for capturing scenarios where transmission sequences are normal but they violate access rights (maintained in system state). For brevity, we omit the description of extension (2) here.

Let Σ be the finite set of events, ε the null event, *ev* a variable of some type of event, *x* an event parameter ranging over all possible values in the domain of *ev*, and *E* an expression. Atomic expressions have one of the following forms: an event *e*, the null event ε, and the parameterized event *ev=x*. Expressions have one of the following forms: *E**, which denotes 0 or more occurrences of *E*; E^+, which denote 1 or more occurrences of *E*; E^K, which denotes *K* occurrences of *E*; *E*; *F*, which denotes *E* followed by *F*; and *E* + *F*, which denotes nondeterministic choice *E* or *F*. The parameterized event *ev=x* abbreviates nondeterministic choice over an event of type *ev*. The event parameter *x* is scoped: multiple occurrences of *ev=x* within the scope of *x* in an expression must be matched with the same choice; in other words, *x* is existentially quantified over its scope. (Unless explicitly specified, the scope of an event parameter is the entire expression.) For example, if Σ={*a*,*b*,*c*} and the domain of *ev* is {*a*,*b*} then the expression ((*ev=x*);c;(*ev=x*)*) abbreviates (a;c;a*) + (b;c;b*). Lastly, as a convenience, the symbol Σ is overloaded to abbreviate the expression that denotes a don't-care event, i.e., a nondeterministic choice over all events in Σ; thus, Σ* denotes 0 or more occurrences of events.

In terms of this language, letting Σ denote the events that can occur on the powerline, the first pattern reduces to witnessing on the powerline the following regular expression within ΔS time:

$$\Sigma^* ; (Function\_Command=x)^K ; \Sigma^*$$

where *x* is a parameter that ranges over the set of function command values and *Function_Command=x* is a parameter expression that matches events which are function commands *x*. The prefix and suffix Σ* subexpressions capture the normal transmissions that may occur within the ΔS time interval before and after the bad pattern, which consists of *K* identical function commands transmitted consecutively. They also capture the case where more than *K* identical function commands are transmitted consecutively. As another example, the second pattern reduces to witnessing on the powerline within ΔT time:

$$(\Sigma^* ; ((\exists y\ (Address\_Command\_Unitcode=y)^M)+ (\exists z\ (Function\_Command=z)^N)))^L ; \Sigma^*$$

where *Address_Command_Unitcode=y* is an expression that matches events which are address commands with unit code *y* and (∃*y* …) denotes the scope of parameter *y*.

6. Summary and Future Work

In this paper, we have focused on the home networking dependability challenges created by the heterogeneity of network entities and the undependable nature of consumer devices for powerline control. We described the use of the soft-state store as a shared heartbeat infrastructure for building robust lookup services that provide eventual consistency in the presence of device/object connecting, disconnecting, and failing. We implemented an Aladdin Device Adapter as an add-on module to allow existing devices to robustly join and leave the system. We described a system architecture that utilizes private powerline networks and exploits the phoneline network to combat the dependability problems in powerline control. Based on actual experiences, we argued for the need to monitor the powerline for unusual activities that could be generated by non-fail-stop devices, radio interferences, and intruders. We described the implementation of a monitoring tool for detecting abnormal powerline command patterns.

Future work includes adding persistence support for soft-states with low-refresh rates in order to enhance data availability upon system failures; modeling all legal powerline command patterns and designing a comprehensive monitoring tool for detecting all bad patterns; and designing a dependability framework to simplify the implementations of dependability solutions and to facilitate the proofs of end-to-end system stabilization [AP95].

References

[AP95] A. Arora and D. Poduska, "A Timing-based Schema for Stabilizing Information Exchange in Networks," in *Proc. Int. Conf. on Computer Networks*, 1995.

[E99] W. K. Edwards, "Core Jini", Prentice-Hall Inc., 1999.

[Ev96] G. Evans, "The CEBus Standard User's Guide", http://www.cebus.com/training.htm#book, May 1996.

[H98] The Home Phoneline Networking Alliance, "Simple, High-Speed Ethernet Technology for the Home," http://www.homepna.org/docs/wp1.pdf, 1998.

[K94] S. Kumar and E. H. Spafford, "A Pattern Matching Model for Misuse Intrusion Detection," In *Proc. National Computer Security Conference*, pp. 11-21, Oct. 1994.

[S] SIEFAST User Guide, http://www.cis.ohio-state.edu/siefast.

[S98] Silent Servant Home Control Inc., "Automated Home Control," 1998.

[WRA00] Y. M. Wang, W. Russell, and A. Arora, "A Toolkit for Building Dependable and Extensible Home Networking Applications," to appear in *Proc. USENIX Windows Systems Symp.*, Aug. 2000.