Security-Aware and Resilient Architecture of Energy Cyber-Physical Systems

Shane Clark, Partha Pal, Rick Schantz
{sclark, ppal, rschantz}@bbn.com
BBN Technologies

Energy cyber-physical systems (CPSs) present an interesting role-reversal from traditional information systems and networks. Unlike traditional systems that consume energy to process information, energy CPSs collect and process information in support of the efficient generation, storage and delivery of energy. This unique paradigm (see Figure 1) makes energy CPSs a key piece of infrastructure—it is well-accepted that almost all sectors of a modern nation will be severely impacted in the event of energy CPS compromise.

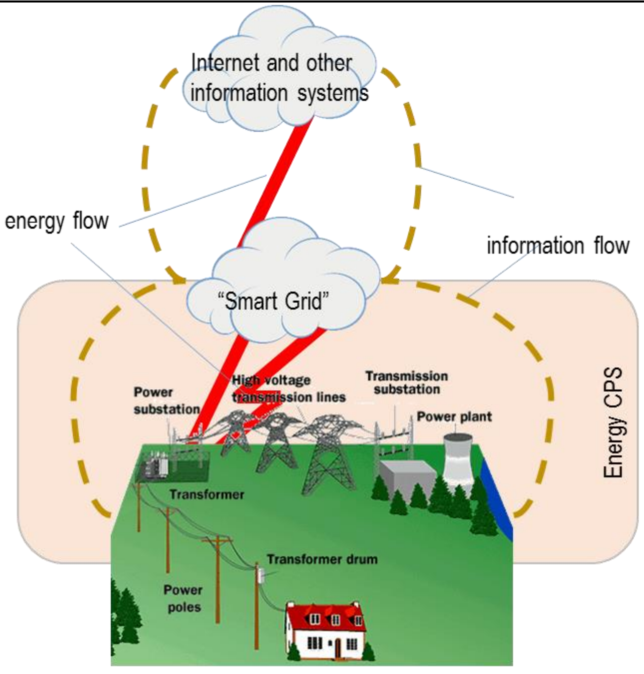

Figure 1: Interaction and interference between energy CPS and traditional information systems

As different segments of the energy CPS take shape in the form of various smart grid subsystems such as SCADA networks, distribution management systems, demand management systems etc., effort has been underway to build in security controls from the ground up [1]. At the same time, economic pressure is pushing towards the use of COTS hardware and software platforms and sharing of existing network infrastructure, often the Internet, in the implementation and roll out of the energy CPS. The reality is that high capital and operational expenses often eliminate designs with stronger security controls from consideration, and the attractiveness of simplified system management and/or vendor support often leads to a hardware and software monoculture.

Based on our experience in the R&D and experimental evaluation of survivability and resiliency techniques for traditional information systems and networks, we take the position that there are a number of existing techniques that should be considered in the design and development of energy CPS. These techniques, which stem from various DARPA- and AFRL-funded R&D efforts, are becoming cost-effective to deploy and operate as they mature into stable and transition-ready solutions. In this paper we will outline a few, and, explain at a high level, their applicability and usefulness in the energy CPS.

The first technique is the addition of *crumple zones*. As Stuxnet and other worms have demonstrated, one cannot rely on “air gaps”—even though there is no network connection, operational needs to access removable media can be abused as a means to propagate. Furthermore, modern hosts often rely on automatic software updates, and manufacturers commonly build in means to establish connectivity for support purposes. We have developed and demonstrated software crumple zones, which protect the boundaries of interacting subsystems with differing levels of trust and the channels through which an application interacts with the network or storage systems [2,3]. At a high level, the crumple zone is a non-bypassable interposition layer designed to absorb attacks and attack-induced failures before they cross critical boundaries. Consider the case of USB-based attack propagation, where the critical application will run in a container (such as a VM) and the crumple zone will be another container (e.g., VM) which will mount the USB device; and any transfer of data from the USB device to the application container will (a) have to be a conscious action taken by an operator and (b) subject to security policies inspecting and controlling what data can be moved. In the case of an attack, it is the crumple zone that will be compromised first. Energy CPSs should have crumple zones at the endpoints of every critical subsystem that interacts, or can potentially interact with, other subsystems (see Figure 2).

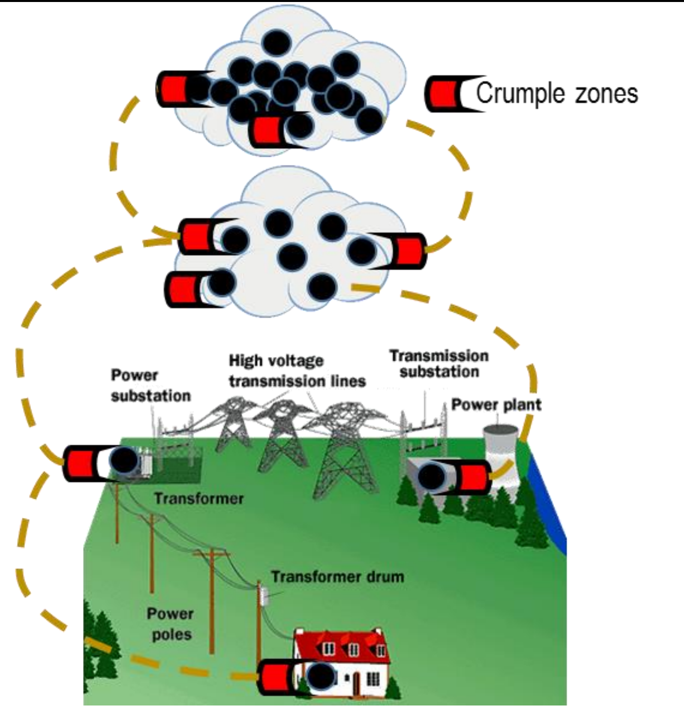

Figure 2: Crumple zone protection known and potential touch points within a system of system embedding energy CPS

The second technique is the intentional creation of artificial diversity as an antidote to common-mode failures and attacks that can wreak havoc in a homogeneous system or monoculture. It is well-established that n-version programming may not be worth its cost [4], but emerging *multicompiler* techniques [5] have been shown to generate a seemingly infinite number of functionally equivalent binaries that have statistically different vulnerability profiles at a very low cost (just the time to compile). This technique is especially attractive for vendor-provided applications (as opposed to say, compiling the entire Linux kernel) that are deployed on consumer-side devices such as set-top boxes or smart meters—each customer’s application will have a unique footprint and vulnerability signature, implying that no single attack is likely to enjoy widespread success. The use of artificial diversity in strategic portions of the energy CPS (see Figure 3) will improve the robustness and resilience of the end-to-end system.

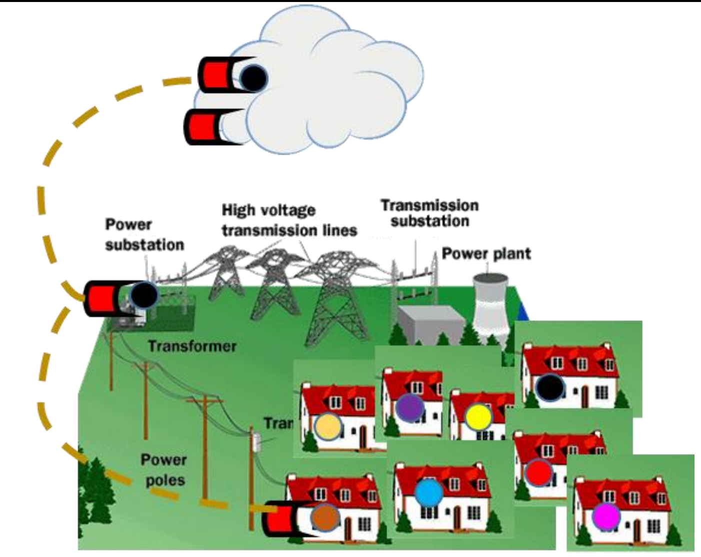

Figure 3: Each consumer device is different—affordable artificial diversity in energy CPS

The third technique is *moving target defense* (MTD), which has evolved from näive port and IP address hopping to techniques that change how a program is laid out and executed. Dovetailing the previous example of artificial diversity in vendor-provided applications, a vendor can update each consumer's application at random intervals with a new variant. Advances in virtualization technology make it possible to modify the fetch-execute cycle of program execution, which in turn offers the possibility of randomly changing how a given binary is handled. While artificial diversity offers a level of spatial containment in the architecture, MTD offers a level of temporal containment—any assumption that an adversary makes about how an application binary is handled at a host is valid only for a random interval. The time to design and incorporate MTD support capabilities like the ability to accommodate dynamic changes and the associated "MTD control plane" into the energy CPS is now.

Even though these techniques are maturing and becoming cost effective, they all involve *adaptation* i.e., the behavior and/or configuration of the application or the protected system changes dynamically—be it a crumple zone stopping the propagation of a message or command or the installation of a new variant. The R&D challenge for such an approach is to make sure that these adaptations are beneficial and not self-defeating—either by establishing a formal proof or an acceptable certification, or by using auxiliary mechanisms to quickly detect and arrest harmful adaptations. Many of these techniques will involve a runtime component and a control plane for managing the adaptation, which is treated as part of the TCB—another R&D challenge is to establish that this is a valid assumption, and that such runtime components can indeed be built as a TCB anchored with an appropriate root of trust.

References

1. Office of the National Coordinator for Smart Grid Interoperability Engineering Laboratory, "NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 2.0," Feb. 2012.
2. P. Pal et al., "Advanced Adaptive Environment – Initial Experience," In Proc. Middleware, Dec. 2011.
3. M. Atighetchi et al., "Crumple Zones: Absorbing Attack Effects Before They Become a Problem," Journal of Defense Software Engineering, Special Issue on Rugged Software, Apr. 2011.
4. J. C. Knight and N. G. Leveson. "An experimental evaluation of the assumption of independence in multiversion programming," IEEE Trans. Software. Eng., Jan. 1986.
5. P. Larsen, S. Brunthaler, and M. Franz, "Security through Diversity: Are We There Yet?," IEEE Security & Privacy, Oct. 2013.