Intrusion-Tolerant Self-Healing Devices for Critical Infrastructure Protection *

Paulo Sousa, Alysson N. Bessani, Wagner S. Dantas, Fábio Souto, Miguel Correia, Nuno F. Neves
University of Lisboa, Faculdade de Ciências, LASIGE – Portugal
{*pjsousa, bessani*}*@di.fc.ul.pt,* {*wagners, fsouto*}*@lasige.di.fc.ul.pt,* {*mpc, nuno*}*@di.fc.ul.pt*

Abstract

Critical infrastructures like the power grid are essentially physical processes controlled by electronic devices. In the last decades, these electronic devices started to be controlled remotely through commodity computers, often directly or indirectly connected to the Internet. Therefore, many of these systems are currently exposed to threats similar to those endured by normal computer-based networks on the Internet, but the impact of failure of the former can be much higher to society. This paper presents a demonstration of a family of protection devices for critical information infrastructures developed in the context of the EU CRUTIAL project. These devices, called CRUTIAL Information Switches (CIS), enforce sophisticated access control policies of incoming/outgoing traffic, and are themselves designed with a range of different levels of intrusion-tolerance and self-healing, to serve different resilience requirements.

1. Introduction

Critical infrastructures (CI) like the *power grid* are essentially physical processes controlled by electronic devices. In recent decades, these electronic devices started to be controlled remotely with off-the-shelf computers connected by common network technologies, often directly or indirectly connected to the Internet [6, 7, 8]. Therefore, today many of those systems that are critical to our society are exposed to a level of threat similar to other Internet services, which are constantly plagued with cyber-attacks. This situation has been recognized by governments and industry, who are promoting research, standards and guidelines to address the problem (e.g., [11, 14]). Although there is considerable reluctance about disclosing information about attacks and intrusions, there are already news about some incidents [6, 12], and much speculation about many others. At least one tool specifically designed to find vulnerabilities in these infrastructures is known to exist [4].

In the context of the EU-IST CRUTIAL project[1], we recently proposed a reference architecture for protecting critical infrastructures [17]. In our opinion, this problem has to be solved at an architectural level mainly because its complexity derives from the hybrid composition of several infrastructures [6, 7, 8]: (1) the operational network, usually called SCADA or PCS[2], contains computers and electronic devices that monitor and actuate on the physical processes (e.g., electricity generation, transformation and transmission); (2) the corporate intranet contains the usual enterprise services (email, databases, etc.) and workstations of personnel, such as engineers who access the SCADA/PCS systems through ad-hoc interconnections; (3) the Internet, through which CI users get to other intranets (e.g., business partners, regulators) and/or simply to the outside world.

The interconnection of these three infrastructures leads to an undesirable link between the Internet and the SCADA/PCS networks, facilitating the propagation of attacks from any place in the world to the control systems. Additionally, we believe that CI protection is more complex than classical network security for other reasons: first, CIs feature a lot of legacy subsystems and non-computer-standard components (controllers, sensors, actuators, etc.), which were deployed when security was not a major concern; second, conventional security practices when directly applied to control devices sometimes stand in the way of their effective operation. These two characteristics imply that we should be very careful when proposing new protection mechanisms, since in most scenarios they have to be implemented without changing the existing SCADA/PCS systems, at least in the medium-term future, and they can not interfere with system operation, specially in emergency situations.

The approach being followed in CRUTIAL is based on securing the interconnections between the different realms of the infrastructures using *CRUTIAL Information Switches* (CIS) [17]. In a nutshell, CIS are protection devices that

*Supported by the EU through project IST-4-027513-STP (CRUTIAL) and the FCT through the Multiannual and CMU-Portugal Programmes.

[1] Critical UTility InfrastructurAL Resilience: `http://crutial.cesiricerca.it`.

[2] Supervisory Control and Data Acquisition / Process Control System.

enforce *sophisticated access control policies on incoming/outgoing traffic*. Our point is that interference and attacks start at the level of the macroscopic data flows between these realms, so protecting these data flows using proper access control policies is a fundamental step towards security. Furthermore, these policies must be enforced by highly dependable devices, the CIS, which are designed with a range of different levels of *intrusion-tolerance and self-healing* to serve distinct resilience requirements.

The paper describes a demonstration of a remotely managed power generation control scenario. It includes a number of computers that emulate both a power generation infrastructure and attacks coming from the Internet, and several prototypes of the CIS design. The main objectives are: (1) to illustrate what kind of attacks can compromise current power system facilities protected by traditional firewalls; (2) to present a family of mechanisms that can be employed to make protection devices incrementally more resilient; and (3) to show that a CIS can offer a more complete and secure solution for the protection of critical systems than traditional firewalls.

2. CRUTIAL Information Switches

In CRUTIAL, the interconnection of realms of a critical infrastructure is modeled as a WAN-of-LANs [17] – realms correspond to a LAN, and LANs are connected by a WAN. Connections of LANs to the WAN are secured by different CIS, which enforce access control policies on incoming and outgoing messages. This could suggest that CIS are simply firewalls. Although they share, of course, some common features, CIS are in fact more than traditional firewalls, and the differences are fundamentally two:

First, although CIS are devices that are placed at network boundaries (just like firewalls), they play a global role of enforcing access control policies of macroscopic data flows among realms of a CI or interconnected infrastructures. Therefore, policies are not expressed simply as a set of local rules, but globally using an *organization-based access control model*, PolyOrBAC [9], capable of expressing policies involving several organizations (e.g., CIs related to the production, transmission, distribution and regulation of the power grid). In this sense, CIS are also more alike to distributed firewalls [1] than to traditional firewalls, and to application-level firewalls than to packet-filters.

Second, the criticality of the infrastructures that we are considering requires protection devices much more resilient than traditional firewalls, which are known to have vulnerabilities [10][3]. To give the CI designers a tradeoff with cost and complexity, we have proposed *a family of increasingly more resilient CIS*, where the strongest devices of the family are both intrusion-tolerant and self-healing.

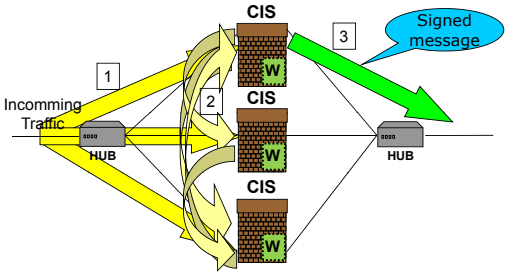

Figure 1. The ITCIS architecture.

In the rest of this section, we briefly describe the CIS instantiations. More technical details about these devices can be found at [2, 3, 13, 16].

Intrusion-Tolerant CIS The intrusion-tolerant CIS (ITCIS) is replicated in a set of computers in order to mask intrusions in some of its components. More precisely, the CIS access control functionality is replicated across $2f + 1$ machines to tolerate up to f malicious or accidental faults. A replica in which there is an intrusion or a crash is said to be faulty.

An ITCIS representation for $f = 1$ is shown in Fig. 1. The ITCIS is composed by three replicas/computers and two hubs (i.e., devices that broadcast the traffic they receive in one port to the other ports). The idea of intrusion tolerance is that if up to f (1 in this case) of the replicas in the figure are attacked, intruded and entirely controlled by the attacker, the ITCIS will still perform its service correctly. For this approach to make sense, replicas have to be diverse, i.e., they should have at least different operating systems and software. This need of diversity limits the maximum number of replicas. For example, the number of diverse operating systems for PCs and PC-like servers is reasonably small.

For the sake of simplicity we explain how ITCIS works considering only the incoming message flows, because for outgoing traffic the explanation is identical. When a packet arrives to the ITCIS:

1. it is forwarded by the left hub to the replicas (step (1) in the figure);
2. each replica checks if the packet satisfies the security policy of the organization (specified using PolyOrBAC);
3. if $f + 1$ replicas vote in favor of accepting the packet (step 2), the packet is forwarded by the current leader replica (step 3), otherwise it is discarded.

This basic mechanism raises two issues. First, how do we prevent a faulty replica from forwarding a packet that

[3]For example, the numbers of serious vulnerabilities in commercial firewalls that allowed *intrusions*, reported by the National Vulnerability Database for 2005, 2006, and 2007, were respectively 9, 15, and 15. The vulnerabilities that allowed *denial of service attacks*, which have an important impact on CIs settings due to timeliness requirements, were respectively 11, 8, and 21.

does not satisfy the policy? The solution is based on the following idea: we consider that traffic forwarded by the CIS must follow the IPSec/AH protocol, and that the key K used to sign the packets with a MAC is stored in a secure subsystem (generically called a *wormhole* [15]) inside each replica; a wormhole only returns a signature for a packet if the replica shows that $f + 1$ replicas gave their approval, by providing the corresponding votes. ITCIS is said to be intrusion-tolerant because any of the replicas can fail, albeit only up to f, and the system still behaves correctly. On the contrary, the wormhole can not fail maliciously for the system to behave correctly (although it can crash). More precisely, the wormhole has to be constructed in such a way that it is not possible for the attacker to tamper with the operation of the wormhole (integrity) or disclose the keys stored inside it (confidentiality). This involves a careful design process, possibly with an implementation in hardware (like in a Trusted Platform Module) or using an hypervisor and virtual machines (which is what we use in the current prototype).

Second, how do we deal with a faulty leader that does not forward a packet that satisfies the policy? To address this problem, all replicas monitor what is transmitted by the leader (the hub on the right hand side of the figure broadcasts the packets back to all); when a leader is detected to behave erroneously, an election protocol runs to nominate a new leader.

Overall, the extra resilience of the ITCIS comes from the fact that it is impossible to subvert correct operation even if there are intrusions in f replicas. Notice for example that a traditional firewall can be undermined with a single intrusion, so the ITCIS is far more resilient. There is, however, still a threat against the ITCIS: given enough time, an attacker may be able to take control of more than f machines, and control the protection system. This problem is solved by the next CIS design.

Intrusion-Tolerant CIS with Proactive Recovery The ITCIS with proactive recovery (ITCIS-PR) works basically as the ITCIS, but has a *self-healing* capability. This capability is implemented by recovering (or rejuvenating) periodically each replica to remove the effects of any intrusion that might have occurred. The recovery procedure involves four tasks: (1) shutdown of the CIS replica; (2) selection of a clean system image for the replica; (3) copying the system image to the replica file-system; (4) booting the new system. In order to ensure the timeliness and correct execution of these actions, recoveries are managed by the secure wormholes, which enforce correctness of the process even under attack. Additionally, the ITCIS-PR needs more replicas to guarantee that system availability is preserved during the recovery of replicas (at least $2f + k + 1$, where k is the maximum that can recover at the same time). Intrusions in replicas are the result of the combination of attacks and vulnerabilities, so the latter should be removed or at least modified when a recovery is done. This is an issue that requires further research but that can be tackled today using mechanisms like memory layout randomization [18]. Most attacks that allow the execution of arbitrary code in the victim machine are done using buffer overflow attacks that require some knowledge about the organization of the memory of that machine (e.g., of the an address of the *libc* to which the attacker wants to force a jump, in the case of an arc-injection attack). Memory layout randomization makes these attacks extremely difficult by changing how the memory is organized, i.e., the (virtual) memory addresses where the application code, DLLs and other third-party software are stored whenever the machine reboots.

ITCIS-PR is perpetually-resilient in the sense that it tolerates intrusions that may occur indefinitely during the system lifetime. However, from the moment of a successful replica intrusion and until its total rejuvenation, the faulty replica can send malicious packets to the inside systems, trying to find and exploit some (known or unknown) vulnerability. Notice that these vulnerabilities do exist and that they are similar to vulnerabilities in Internet systems. The US-CERT lists several buffer overflows, authentication problems and improper message handling vulnerabilities in products from several control systems manufacturers[4]. This problem is solved by the next CIS design.

Intrusion-Tolerant CIS with Proactive and Reactive Recovery This CIS (ITCIS-PRR) is similar to ITCIS-PR but adds another protection layer. Besides the periodic rejuvenation of replicas, each replica monitors the behavior of all others (for example, by looking at the voting decisions and the packets forwarded by the leader). If a set of replicas discover that another one is misbehaving, they force the recovery of this replica with the assistance of the secure wormhole [13]. This wormhole provides a service that schedules both types of recoveries (periodic and due to fault detections) in order to maintain the availability of the CIS. It uses a private network to exchange information and support coordinated actions.

CIS Performance and Resilience Above we described the differences and benefits that each CIS instantiation brings. However, we want to quantify these benefits in terms of percentage of failed time of each of these instances (and a traditional firewall) in unattended missions. This assessment was made using the Mobius tool and a summary is the following [16]. We consider that the mission time is 10,000 hours (approximately 1 year) and we varied hosts' minimum inter-failure time (*mift*) from 1 to 1000 hours. For

[4]See `http://www.us-cert.gov/control_systems/csdocuments.html#vuls`

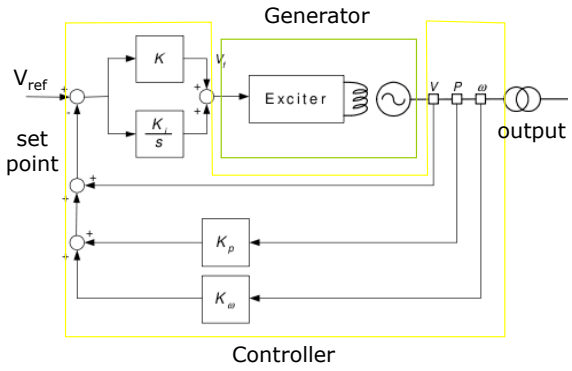

Figure 2. Automatic voltage regulator (AVR).

a normal non-replicated firewall, the percentage of failed time varied from about 90% with *mift* of 1000 hours, to 100% with *mift* of 1 hour. For ITCIS, the numbers were similar but lower: went from 70% to 100%. For ITCIS-PR and ITCIS-PRR with a recovery period of 10 minutes, which is the one of the current prototype, the percentage of failed time is almost zero, for all values of *mift*.

Another aspect that we evaluated is the overhead introduced by the CIS, i.e., the delay it introduces in the communication (see [5]). In normal conditions, all instantiations of the CIS introduced a delay from 0.5ms to 2ms. We also emulated what would happen under a denial of service attack by generating DoS traffic with *iperf*. Until 70Mbps of traffic, the delay remained below 4ms. With 100Mbps the delay increased to 14ms.

3. A Power Generation Scenario

The scenario we consider is a remotely controlled electricity generation system. A power system has to regulate several variables, like voltage, frequency, and active and reactive powers. However, to simplify the following discussion, we will concentrate on regulating a single variable, the *voltage*, and the objective is to maintain a correct voltage despite cyber-attacks.

Project CRUTIAL presented a set of power control scenarios [7]. One scenario that is already in use in some countries comprises a *hierarchical voltage regulation*. The system is supposed to have a national dimension and it includes four main components: (1) National Voltage Regulator (NVR): collects data from the field and decides which are the voltages to be achieved by each node of the power grid to attain a global optimal voltage pattern. (2) Regional Voltage Regulator (RVR): defines the reactive power for each power plant in its region. (3) Reactive Power Regulator (PQR): based on the reactive power defined by its RVR, the PQR defines which are the voltage levels that must be generated by each one of its automatic voltage regulators. (4) Automatic Voltage Regulator (AVR): automatic controller

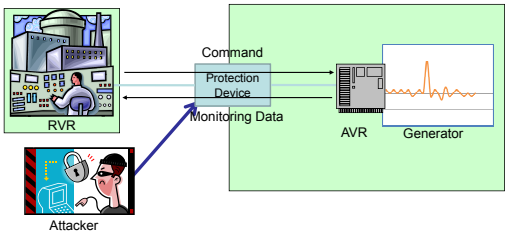

Figure 3. Attack to the power regulation.

attached to the generator, which implements a control rule to drive the generator towards a set point defined by its PQR.

For simplicity, we consider *only one RVR controlling one AVR* (we disregard the NVR, and since there is a single AVR, we also ignore the PQR). The RVR defines the voltage level set points and the AVR drives the generator towards these set points. The controlled variable is the generator *output voltage*. Fig. 2 details the structure of an AVR and the generator, together representing a classical PI (Proportional-Integral) control loop. The input is the voltage set point V_{ref} set by the RVR.

Attack Scenario The global attack scenario is depicted in Fig. 3. The RVR is in some place outside the power plant and accesses the AVR using a network connection. The communication between the RVR and the AVR consists essentially in two kinds of messages: the RVR sends set point commands to change the output voltage of the generator, and gets periodic monitoring information and voltage levels from the AVR. The LAN where the AVR is located is secured from outside threats using a protection device, which can be either a traditional firewall or one of the CIS designs. The attacker attempts to disrupt the correct behavior of the generator by bypassing the protection device.

A first kind of vulnerabilities that concern us are those that allow the attacker to compromise the protection device itself. A typical case consists in finding a buffer overflow vulnerability in the protection device, then executing that attack gaining access to the machine, and finally modifying its behavior (e.g., switching off the protection or sending directly packets to the controller). This last stage can involve doing a privilege escalation attack, but that is usually simpler than gaining access to the machine. Over the years several types of remotely exploitable vulnerabilities have been discovered. One example is the Symantec Norton Firewall Buffer Overflow (Bugtraq ID 5237-2002) that lets *"the attacker to execute code illegitimately"* in the firewall computer. Another kind of vulnerabilities that concern us are those that allow the attacker to bypass the protection mechanism. One example is the FreeBSD IPFW Filtering Evasion Vulnerability (Bugtraq ID 2293-2001) that *"allows an attacker to possibly send illegitimate packets to a protected host"*.

If at some point in time, an attacker is able to communicate with the controller, he/she can do one of two things:

- Define target set points arbitrarily.
- Attack the controller host trying to explore vulnerabilities in its software. Once the controller is compromised, the attacker can change most of the control parameters (e.g., K_p, K_ω from Fig. 2) at will.

In both cases, the attacker can affect the controllability of the generator and potentially provoke some physical damage on it and disturb the power grid.

4. The Demonstration

The demonstration uses the scenario described in the previous section to give evidence of the expected resilience of four protection devices that can be deployed on critical infrastructures: traditional firewall and the 3 CIS versions.

The setup of the demonstration is depicted in Fig. 4. It involves 7 computers and 2 switches/hubs. The computers represent the following entities: an emulator of the RVR and the attacker (1 computer each), a traditional firewall or the CIS (1 to 4 computers), and an emulator of the AVR and generator (1 computer). The displays from various software applications are used to show the voltage levels defined and perceived by the RVR, the output voltage of the AVR/generator, and the internal state of the protection device (e.g., intrusion in replica II and replica IV recovering).

The demonstration has four phases, each one for a different protection device. Every phase illustrates basically two things: (a) the device effectively protecting the AVR; (b) an example of how the protection device can fail and how this affects the AVR.

Traditional firewall In the first phase, the protection device is a firewall. First the attacker console (bottom left of Fig. 4) is used to emulate the attacker issuing some malicious commands, but the firewall drops these packets because they do not satisfy the access control policy of the infrastructure, as expressed by the firewall rules. The input voltage level of the emulator of the AVR/generator (right hand side of Fig. 4) is not affected and the output voltage stays stable. Then, we explain how the firewall can fail due to a vulnerability (basically the discussion made in Section 3), and how this might allow the attacker to execute commands that affect the AVR, such as defining a dangerous voltage set point. For this purpose, the two above mentioned vulnerabilities (Bugtraq ID 5237-2002 and 2293-2001) are explained and an attacker that performs these attacks is emulated, once more using the attacker console. In both cases the attacker ends up being able to send packets to the AVR and we show how he/she is able to modify the voltage level in the emulator of the AVR/generator.

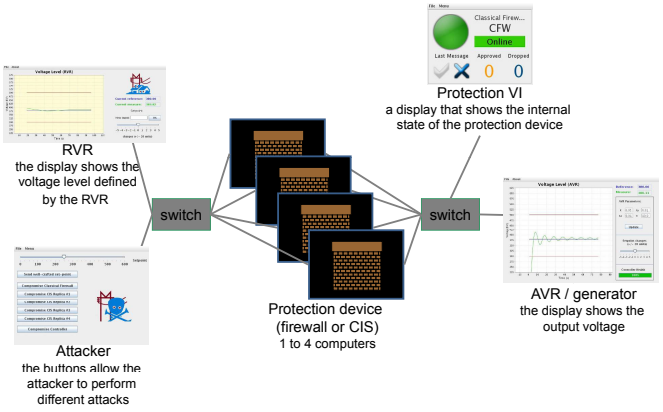

Figure 4. The setup of the demonstration, including some example displays.

ITCIS In the second phase, the protection device is an ITCIS with 3 replicas (tolerates 1 intrusion, i.e., 1 faulty replica). First, we illustrate the application working normally like before (stable input and output voltage even if attackers send commands that are discarded by the CIS). Next, we use the attacker console to emulate an intrusion in one of the replicas, then use this position to send commands to the AVR. The demonstration shows that the AVR does not accept these commands because the faulty CIS replica is unable to sign the message with a valid MAC (it does not have access to the secret key K). Finally, we emulate a successful attack to another CIS replica, showing that the control of a majority of replicas allows the attacker to produce valid MACs, thus send control commands to the AVR and setup the voltage levels he/she wants.

ITCIS-PR In the third phase, the protection device is a ITCIS-PR with four replicas. This configuration sustains one intrusion every 10 minutes, and to ensure the availability of the CIS, at most one replica can be rejuvenated at each moment. The interval of 10 minutes comes from the time needed to recover a replica, which is 2.5 minutes in our prototype (with SATA disks and 1.7GByte Linux system images). This window of vulnerability could be reduced substantially with a more optimized setup, for example, with solid state disks and smaller system images (less than 500MBytes).

In this phase, we show that the ITCIS-PR prevents the adversary from controlling the AVR, even if he/she manages to compromise two replicas, as long as the intrusions happen in different rejuvenation intervals (i.e., at most every 10 minutes). The attacks are performed identically as for ITCIS. The display with the internal state of the protection device (top right of Fig. 4) is quite useful for this demonstration because it allows us to watch when replicas are recovering or have intrusions. This phase finishes with a faulty replica executing a vulnerability scan against the

AVR between recoveries (once more this request is issued in the attacker's console).

ITCIS-PRR The fourth phase uses the ITCIS-PRR, which is similar to the ITCIS-PR, but the CIS replicas also monitor each other behavior and force a reactive recovery when a replica sends erroneous packets. The demonstration is similar to the one of ITCIS-PR but also shows that when a faulty replica starts to perform a network scan, it is almost immediately forced to rejuvenate, so the network scan attack stops almost after it starts.

5. Conclusions and Discussion

The demonstration described in this paper illustrates the kinds of attacks that can compromise power system facilities protected by traditional firewalls, and presents a family of mechanisms that can be employed to make a protection device incrementally more resilient (ITCIS, ITCIS-PR, ITCIS-PRR). Furthermore, it shows that the CIS is a device that offers a more complete and secure protection for critical systems than firewalls. With the most resilient of the self-healing intrusion-tolerant devices – ITCIS-PRR – in place, the attacker is seriously constrained. After considerable effort, he/she may eventually be able to compromise one of the replicas, but this replica would soon be rejuvenated, and if in the meantime he/she tries to attack the protected infrastructure, the replica would be recovered immediately.

Interesting questions are the cost, challenges and practicality of adopting the CIS in a real critical infrastructure. In a real environment, a CIS should be placed in the same locations as firewalls [11, 14], i.e., at least in network interconnections (e.g., in the interconnection between a substation and the utility network). Therefore, adding a CIS to a CI is as practical as putting a firewall. The main challenge is also similar: defining access control policies that effectively protect the CI. The cost of individual CIS is clearly higher than the cost of one firewall due to the need of replication. However, the system designers or administrators can choose between doing the replication with physical machines or inside a single machine using virtualization (but loosing the ability to tolerate hardware faults) [2, 3]. Nevertheless, protecting a CI is not a luxury but a need, and its cost has to be balanced with the impact that a security compromise might have to society (e.g., having a generator destroyed or a blackout in a certain region).

References

[1] S. M. Bellovin. Distributed firewalls. *;login:*, Nov. 1999.

[2] A. N. Bessani, P. Sousa, M. Correia, N. F. Neves, and P. Verissimo. Intrusion-tolerant protection for critical infrastructures. DI/FCUL TR 07–8, Department of Informatics, University of Lisbon, April 2007.

[3] A. N. Bessani, P. Sousa, M. Correia, N. F. Neves, and P. Verissimo. The CRUTIAL way of critical infrastructure protection. *IEEE Security & Privacy*, pages 44–51, Nov./Dec. 2008.

[4] G. Devarajan. Unraveling SCADA protocols: Using Sulley fuzzer. presentation at DEFCON-15, Aug. 2007.

[5] S. Donatelli et al. Experimental validation of architectural solutions. Project CRUTIAL EC IST-FP6-STREP 027513 Deliverable D20, Mar. 2009.

[6] D. Dzung, M. Naedele, T. P. V. Hoff, and M. Crevatin. Security for industrial communication systems. *Proceedings of the IEEE*, 93(6):1152–1177, June 2005.

[7] F. Garrone et al. Analysis of new control applications. Project CRUTIAL EC IST-FP6-STREP 027513 Deliverable D2, Jan. 2007.

[8] V. M. Igure, S. A. Laughter, and R. D. Williams. Security issues in SCADA networks. *Computers & Security*, 25:498–506, 2006.

[9] A. E. Kalam, Y. Deswarte, A. Baina, and M. Kaaniche. Access control for collaborative systems: A web services based approach. In *Proceedings of the IEEE International Conference on Web Services*, pages 1064–1071, 2007.

[10] S. Kamara, S. Fahmy, E. Schultz, F. Kerschbaum, and M. Frantzen. Analysis of vulnerabilities in Internet firewalls. *Computers and Security*, 22(3):214–232, Apr. 2003.

[11] President's Critical Infrastructure Protection Board and Office of Energy Assurance U.S. Department of Energy. *21 Steps to Improve Cyber Security of SCADA Networks*. U.S. Department of Energy, 2002.

[12] R. Schainker, J. Douglas, and T. Kropp. Electric utility responses to grid security issues. *IEEE Power & Energy Magazine*, Mar./Apr. 2006.

[13] P. Sousa, A. N. Bessani, M. Correia, N. F. Neves, and P. Verissimo. Resilient intrusion tolerance through proactive and reactive recovery. In *Proceedings of the 13th IEEE Pacific Rim Dependable Computing Conference*, pages 373–380, Dec. 2007.

[14] K. Stouffer, J. Falco, and K. Kent. Guide to supervisory control and data acquisition (SCADA) and industrial control systems security. Recommendations of the National Institute of Standards and Technology. Special Publication 800-82, NIST, Sept. 2006. Initial Public Draft.

[15] P. Verissimo. Travelling through wormholes: A new look at distributed systems models. *SIGACT News*, 37(1):66–81, 2006.

[16] P. Verissimo, A. N. Bessani, M. Correia, N. F. Neves, and P. Sousa. Designing modular and redundant cyber architectures for process control: Lessons learned. In *Proceedings of the 42nd Annual Hawaii International Conference on System Sciences*, Jan. 2009.

[17] P. Verissimo, N. F. Neves, and M. Correia. The CRUTIAL reference critical information infrastructure architecture: a blueprint. *International Journal of System of Systems Engineering*, 1(1/2):78–95, 2008.

[18] J. Xu, Z. Kalbarczyk, and R. K. Iyer. Transparent runtime randomization for security. In *Proceedings of the 22nd IEEE Symposium on Reliable Distributed Systems*, pages 260–269, Oct. 2003.