Resilient Metro-Scale Smart Structures: Challenges & Future Directions

Mike Burmester[1] and Jorge Munilla[2]
[1]*Department of Computer Science, Florida State University, Tallahassee, FL 30302, U.S.A.*
[2] *Campus de Excelencia Internacional Andalucia Tech, Universidad de Malaga, Malaga, Spain, 29071.*
burmester@cs.fsu.edu, munilla@ic.uma.es

Keywords: Smart Structures, Resilience, Smart Grids, Supply Chain, Logistics, IoT.

Abstract: Smart structures are highly inter-connected adaptive systems that are coordinated by cyber systems to optimize specific system objectives. In this paper we consider the challenges for securing metro-scale smart structures. We use a threat model that allows for untrusted behavior to capture realistic IoT scenarios, and discuss vulnerabilities, exploits and attack vectors. Resilience is defined in terms of stability, resistance to damage and self-healing. To illustrate the challenges of capturing resilience we consider two very different applications: supply chain logistics and smart grids. Both are mixed latency and throughput sensitive, each in their own particular way. The first involves scanning RFID tagged objects in pallets. An untrusted RFID reader is given a one-time authenticator to inspect a pallet and identify any missing objects; and, if there are no missing objects, compile a proof of integrity. The reader should not be able to trace objects via unauthorized inspections (privacy). This application uses RS erasure codes that are more appropriate for memory constrained RFID tags. The second application involves securing industrial substation automation systems. These are particularly vulnerable to cyber attacks, and HIL testbeds are used for real-time multilayer vulnerability analysis. For metro-scale applications we propose virtualized testbeds that are portable and suitable for onsite incidence response. For each application we show how metro-scale analytics are used to capture resiliency.

1 INTRODUCTION

The Internet of Things (IoT) links identifiable objects to their virtual representation on the Internet making it possible for an end user (process) to monitor and link these to additional information regarding their status for efficient control, management and logistics. This extends the scope of the Internet making it possible to control smart systems and structures (RAE, 2012), in particular industrial control systems and critical infrastructures. In this paper we consider the challenges of protecting such applications. Smart/critical infrastructures are a prime target for cyber attacks (The White House, 2013). These may involve nation-state (and ideological) actors, intelligence services, terrorists, industrial spies, criminal groups, hacktivists, hackers (e.g., botnet operators), spyware/malware authors, etc., as well as insiders (ICS-CERT, 2015). Attacks may also be physical, or physical-enabled. Such attacks enable the attacker to penetrate the layered defenses of smart structures. To illustrate the challenges we consider two applications: the first involves a supply chain while the second an electric grid. We show how in both cases metro-scale analytics can be used for resiliency.

The paper is organized as follows. In Section 2 we model smart structures as tightly coupled ecologies and define resilience in terms of survivability and self-healing. We then consider two applications: a supply chain in Section 3 and an electric grid in Section 4, and show how to capture resilience. We conclude in Section 5.

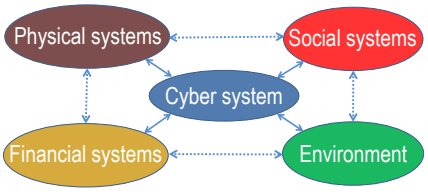

Figure 1: An infrastructure ecology.

2 SMART STRUCTURES

Smart structures are highly inter-dependent systems that integrate a tightly coupled mixed-latency ecology consisting of physical systems (sensors, embedded devices, etc), social systems (operators, customers) financial systems (procurement/acquisition, etc) and the environment (Miller and Page, 2009), that are coordinated by cyber systems so as to optimize specific system objectives based on their properties and constraints, as well as their current and estimated state (Figure 1). Depending on the application, these structures are called supervisory control and data acquisition systems (SCADA), industrial control systems

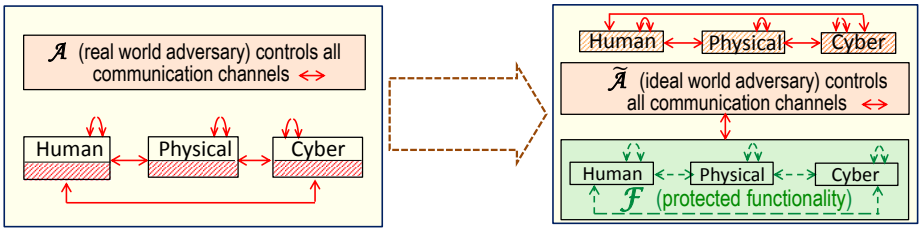

Figure 2: Real vs ideal world simulations.

(ICS), or distributed control systems. Because of the interdependencies, failure in any one of the components may have a ripple (or cascade) effect on others and lead to infrastructure disruption with potentially disastrous impact on the services provided. Interdependencies can be exploited by an adversary. It is well known that dependent applications that run (concurrently) on the same platform can lead to exploits that may compromise the system, *e.g.,* return-oriented programming attacks (Roemer et al., 2012). However cross-domain dependencies may also lead to exploits: Stuxnet used USB flash drives to bridge the air gap protection of a SCADA system (Langer, 2011).

Due to the potential catastrophic impact of infrastructure failure on the services provided, it is essential that mechanisms that support the integrity of operations that monitor and control the system are employed: wrongly received/executed or dropped commands may render the structure unstable (Burmester et al., 2012; Guidry et al., 2012). Protection must ensure continuity of service, requiring *real-time* control, and resist a formidable array of natural and man-made hazards that include bad/faulty design, cyberspace attacks and terrorist acts (Richard G. Little, 2002). In particular, it must resist coordinated hazards. Protection must therefore extend to the components of systems and guarantee that these do not get compromised. This shifts the focus of smart structure protection towards resiliency, accentuating self-healing and survivability behavior, that in turn leads to risk management and threat mitigation.

2.1 Reliability & Resilience

Any attempt to provide holistic protection for highly interdependent smart structures will fail because of their complexity. The best we can aim for is risk management and threat mitigation.

Reliability typically refers to the proper functioning of the structure, as defined by its specifications and policies, and captures fault-tolerance. There are several definitions that describe different aspects of resilience, depending on the application. For engineering systems, resilience requires constancy, predictability and stability near an equilibrium steady state; for social systems, a balance of self-organizing systems; for environmental systems, resistance to damage and quick response to natural perturbations/disturbances: for financial systems, coping with change and adapting to consequences of failure.

2.2 Threat Model

The UC formalization (Beaver, 1989; Canetti, 2001) can be used to analyze the vulnerabilities of interdependent applications and is ideally suited for studying the threats of an infrastructure from a holistic point of view. This models all parties, including adversarial parties, by efficient processes (probabilistic polynomial-time Turing machines) and uses a real world simulation to model the actual behavior of the system in the presence of a malicious (Byzantine) adversary $\mathcal{A}$ (Figure 2, left), and an ideal world simulation to model the protected behavior of the system (Figure 2, right), in which a trusted functionality $\mathcal{F}$ enforces its protection policies—this captures *operational effectiveness*. In the real world the adversary $\mathcal{A}$ controls the communication channels between all parties (Figure 2, left: solid red arrows for non-compromised parties and shaded green boxes for compromised parties.). $\mathcal{A}$ may replay, modify/drop or fabricate messages. In the ideal world the tasks of non-compromised parties (dashed arrows) are executed by the functionality $\mathcal{F}$ that enforces the specifications and policies of the infrastructure, with the adversary replaced by an ideal adversary $\widetilde{\mathcal{A}}$ that emulates $\mathcal{A}$. For UC-security, the two simulations should be indistinguishable by any efficient process (the *environment*).

Although the UC-formalization is too restrictive for resiliency, it can be used to describe and illustrate specific attacks. For example, an *exploit* can be described as a *cause-effect* action $X2Y$ involving $X,Y \in \{H,P,C\}$, $H \in$ Human, $P \in$ Physical and $C \in$ Cyber (Yampolskiy et al., 2013), in which X causes an effect on Y that results in the change of Y's state. Potential exploits are illustrated in Figure 2 by solid red arrows on the left; in the ideal simulation they correspond to dashed green arrows, since

$H2H$	social engineering, impersonation, DOS
$H2P, P2H$	impersonation (H), substitution (P)
$H2C, C2H$	impersonation (H), spoofing(C), phishing(C)
$P2P$	substitution, DOS
$P2C, C2P$	substitution(P),air gap(P),side-channel,DOS
$C2C$	side-channel, timing/power, ROP

Figure 3: Typical exploits of smart structures.

here we get the protection of the trusted functionality $\mathcal{F}$. Such exploits are clearly distinguishable by the environment. Corrupted/compromised entities are in shaded red boxes. The red solid arrows on the right and left in Figure 2 are also exploits (*e.g.*, insider)—these are excluded from the UC formalization.

The adversary may also control some vertices: vertex X may be infected by malware ($X \in C$), contain compromised hardware ($X \in P$), or have deception capabilities ($X \in H$).

2.3 Attack Vectors

The graph G with edges the actions $X2Y$ and vertices in $H \cup P \cup C$ models the vulnerabilities of an infrastructure. The subgraph $G' \subset G$ consisting of the exploits $X2Y$ is the *threat graph*. Any path of G that shares an edge with G' is an *attack vector*. An attack vector typically involves an action in which an adversarial vertex delivers a malicious outcome to a target vertex.

Figure 3 lists some common types of inter- and cross-domain exploits with their corresponding description. $H2H$ exploits involve social engineering; $P2Y$ (or $Y2P$), any Y, involve substitution: a physical object is substituted, damaged or compromised. $C2Y$ (or $Y2C$) exploits involve side-channel attacks that undermine privacy, *e.g.*, leak private key information (Kocher, 1989; Standaert et al., 2009), and return-oriented programming that undermine integrity (Abadi et al., 2009; Roemer et al., 2012). Air gaps (RFC 4949, http://tools.ietf.org/html/rfc4949) are used to ensure physical isolation of critical structures (such as military computer systems, SCADA and ICSs) from unsecured networks. Air gap exploits $C2C$ use removable media holes to bridge the gap—USB flash drives were used by service contractors in the Stuxnet attack (Langer, 2011).

Exploits that are known can be prevented by enforcing protection policies. However zero-day exploits that target previously unknown or undisclosed vulnerabilities cannot be prevented until they get discovered and analyzed. Attack vectors are a means by which the adversary can deliver a payload or malicious outcome to a smart structure, or for exfiltration. Figure 4 illustrates an air-gap attack vector that delivers a malicious outcome: the adversary first uses an $H2C$ exploit to compromise a flash drive and then an $H2H$ social engineering exploit to fool a service contractor that the flash drive is authentic. In the $H2C$ exploit the flash drive is transfered and in the $C2C$ exploit a malicious outcome is delivered.

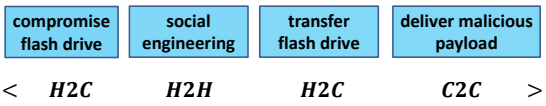

Figure 4: Fragments of an air-gap attack vector.

Threats such as these are a major security concern. In particular advanced persistent threats (APT) that employ $C2P$ exploits using unexpected commands to the programmable logic controllers (PLC) of smart structures (the Stuxnet attack used an infected rootkit to modify codes and send unexpected commands to the PLC while returning a loop of normal operations system values feedback to the user).

2.4 Security Assessment & Testing

This involves developing a security assessment policy and methodology as well as procedures and tools for identifying system vulnerabilities. We refer the reader to the NIST Technical Guide for Security Testing and Assessment (SP800-115, 2008). Here we note that testing must include static and dynamic analysis, white, gray, and black box testing, penetration testing, simulation, and ensuring that the system components or services are genuine. These are intended to uncover unintentional and intentional vulnerabilities including, for example, malicious code, malicious processes, defective software, and counterfeits.

We next consider two smart structure applications that benefit from metro-scale analytics. These involve a supply chain and a smart grid.

3 SUPPLY CHAIN LOGISTICS

RFID (Radio-Frequency Identification) is an emerging wireless technology that stimulated numerous innovative applications in several fields such as inventory control, supply-chain management, and logistics, as well as identify new research challenges and opportunities. A typical RFID deployment has three main components: tags or transponders which are electronic data storage devices attached to objects to be identified, readers or interrogators that manage tag

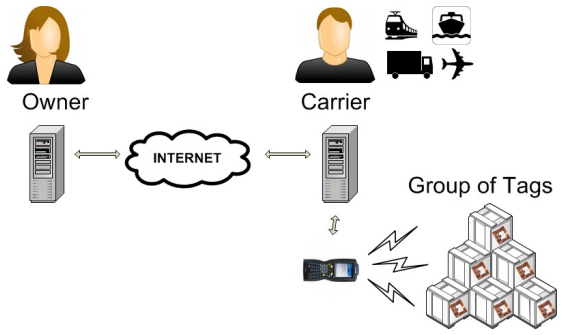

Figure 5: An untrusted Carrier can identify missing objects in a pallet and compile a grouping-proof of integrity when there are no missing objects.

populations and a back-end server (the verifier) that exchanges tag information with the readers and processes data according to specific task applications. Most RFID tags are passive and do not have power of their own but get the energy needed to operate from an RFID reader. Passive tags are inactive until activated by the electromagnetic field generated by a reader tuned to their frequency. Although most RFID applications do not support privacy or integrity, the technology has now found use in many applications that resist privacy and integrity threats. The recently ratified EPC Gen2v2 standard incorporates privacy and integrity mechanisms (EPC-Global, 2015).

3.1 Pallet shipment logistics: integrity & privacy

When RFID technology is used for supply-chain management, concerns regarding the monitoring and transfer of ownership or control of tags have to be addressed. If the transfer is permanent, ownership transfer protocols can be used (Kapoor and Piramuthu, 2012; Munilla et al., 2013). If the owner does not want to cede control, even though this may only be temporal, e.g., if a manufacturer uses a carrier who in turn uses other carriers to transport products, then it is desirable that the owner can periodically check the integrity of a shipment via the carrier(s) (Figure 5). This requirement is referred to as *group scanning* and involves multiple tags generating a *grouping-proof* of "simultaneous" presence in the range of an RFID reader (Liu et al., 2013; Burmester and Munilla, 2013). Below we list some of the most common security requirements for secure group scanning.

(a) The Owner (a trusted entity) can authorize an RFID reader (an untrusted entity) to inspect the pallet and identify any missing tagged objects.

(b) The authorization is for one only inspection, and the tags are untraceable via unauthorised inspections.

(c) The authorized reader can generate a grouping-proof of integrity for a pallet if no tags are missing, while if some tags are missing can identify these.

(d) Only the Owner can verify the grouping-proof.

In the rest of this section we review the literature for RFID grouping scanning, discuss erasure codes, and present an anonymous RFID grouping-proof of integrity with missing tag identification appropriate for metro-scale applications.

3.2 Background

Ari Juels introduced in 2004 the security context of a new RFID application, called a *yoking-proof* (Juels, 2004), that generates evidence of simultaneous presence of two tags in the range of an RFID reader. This protocol was later found to be insecure (Saito and Sakurai, 2005; Juels, 2006) but, group scanning triggered considerable interest in the research community with yoking-proofs extended to: grouping-proofs for multiple tags (Piramuthu, 2006), anonymous grouping-proofs (Burmester et al., 2008; Huang and Ku, 2008; Chien et al., 2009), and grouping-proof for distributed RFID applications with trusted readers (Liu et al., 2013).

While grouping-proofs provide evidence of integrity for complete groups, they do not address incomplete groups, in particular they do not provide any information about missing tags. In 2012 Sato *et al.* proposed a *grouping code* that makes it possible to find the identifiers of all tags of a group including missing tags, without requiring a packaging list or an external database (Sato et al., 2012). This is based on Gallager low-density parity check (LDPC) codes (RFC6816, 2013).

Forward error correction can increase the operating speed and reduce costs when it is difficult to access a database with the corresponding information. However the randomized nature of Gallager LDPC codes makes it difficult to get specific decoding guarantees. To address this issue, several variants were proposed (Su et al., 2013; Su and Wang, 2015; Su and Tonguz, 2013; Su, 2014; Ben Mabrouk and Couderc, 2015). However for these, the size of the blocks and the partitioning of the redundancy is not optimal.

3.3 Erasure Codes

Let $\mathbb{F}_q$ be a finite field of order $q = p^m$, p a prime, m a positive integer. A q-ary (n,k,s) erasure code is a linear forward error correction code that encodes source

(input) data $x = (x_1, \ldots, x_k) \in \mathbb{F}_q^k$ to encoded data $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$, in such a way that the source data can be recovered if no more than s blocks $y_i \in \mathbb{F}_q$ are missing. We must have $s \leq n-k$ (Singelton bound); the optimal case $s=(n-k)$ is reached with Maximum Distance Separable (MDS) codes. The most common MDS codes are the Reed-Solomon (RS) codes that are cyclic over $\mathbb{F}_q$, $q > n$, with minimum distance $d = n-k+1 = s+1$.

In Section 3.4 we shall use an $RS(n,k)$ code over $\mathbb{F}_{2^m}$, $2 \leq m \leq 16$ (based on the operational recommendations of (RFC6865, 2013) for the values of m), to encode the identifiers $(id_1, \ldots, id_{n_g})$ of a collection of n_g RFID tags. For this application the source data $x = id_1 \| \cdots \| id_{n_g}$ is an $n_g \ell$ bit string, where ℓ is the binary length of the identifiers id_i. We rearrange x into k blocks $(x_1, \ldots, x_k) \in \mathbb{F}_{2^m}^k$ (the last block is padded with zeros if necessary). Then x is encoded to get an RS codeword $(y_1, \ldots, y_n)$, with n/n_g blocks stored in the memory of each of the n_g tags tag_i, so as to recover up to $s_t = (n-k)/(n/n_g)$ identifiers of missing tags. The n/n_g blocks of lenght m stored on tag_i are denoted by ID_i and provide the identifying information id_i as well as the redundancy needed to recover missing data.

RS decoding can only be performed if the scanned ID_i are ordered correctly, with gaps for missing values. For this purpose control information is also needed: each ID_i is extended to include some extra bits that define the order i of tag_i when its identifier was encoded. For example, if we use the $RS(150,120)$ code over $GF(2^8)$ for a collection of $n_g = 10$ tags with up to $s_t = 2$ missing tags, then the bit length of ID_i is 124 bits, of which: 96 bits are used for tag identifying data id_i (as recommended by EPC Gen2v2 (EPC-Global, 2015)), 24 bits for the redundancy needed to recover missing tag identification data, and 4 bits for control (Burmester et al., 2016).

3.4 An Anonymous Grouping Proof with Missing Tag Identification

The grouping proof is based on the design requirements in Section 3.1 and provides anonymity. In particular, the tags do not share any private information with the interrogating reader R (an untrusted entity).

Protocol Description

For each group of tags $G = \{tag_1, \ldots, tag_{n_g}\}$ of the owner V, V stores the tuple: $(T_s, K_g, \{(K_i, ID_i)\}_{i=1}^{n_g})$, where T_s is a counter, K_k a group key K_g, and (K_i, ID_i) the private key and identifier of tag_i (Section 3.3). Each tag_i stores in non-volatile memory: (ID_i, K_g, K_i), and a counter T_{s_i} that is initialized to the same value T_s for all tags of G. The reader initially does not share any information with the tags. The protocol is initiated by the owner who sends to the reader R a request (T_s, T_s', K_s), where T_s is a fresh value of a counter, $T_s' = h(K_s, T_s)$ is an authenticator and $K_s = h(K_g, T_s)$ is the session key. The protocol has two rounds—see Figure 6.

1a.	$R \rightarrow *:$ (T_s, T_s')	(set timer)
b.	$tag_i \rightarrow R:$ (r_i, r_i'), $i \in [1..n_g]$	(set timer)
2a.	$R \rightarrow *:$ (R_s, R_s')	(set timer)
b.	$tag_i \rightarrow R:$ $(r_i, M_i, M_i' \oplus y_i, , P_i, P_i')$, $i \in [1..n_g]$	(timeout)
	Grouping-proof: $(T_s, r_1, \ldots, r_{n_g}, h(P_1, \ldots, P_{n_g}))$	

Figure 6: Flows of the anonymous grouping-proof with missing tag identification.

Round 1. R broadcasts to all tags in its range (T_s, T_s') and sets a timer. Each tag_i in range of R, computes $K_s = h(K_g, T_s)$, checks that $T_s' = h(K_s, T_s)$ and verifies that $T_s > T_{s_i}$. If this fails it returns random values. Otherwise it updates the counter to T_s, draws a pseudo-random number r_i and computes its authenticator $r_i' = h(K_s, r_i)$. Then it sends (r_i, r_i') and sets a timer. The received values r_i are used to identify (singulate) tags in this session. For every received r_i, the reader checks its integrity $r_i' = h(K_s, r_i)$. If this is correct, the value r_i is stored as part of the grouping proof. Using these values, R computes a group session challenge $R_s = h(T_s, r_1, \ldots, r_{n_g})$ and its authenticator $R_s' = h(K_s, R_s)$. This round incorporates the randomness provided by the verifier's challenge T_s and the randomness provided by the tags r_i, which prevent replay attacks. The challenge T_s defines the scanning period for the verifier, and the simultaneity by defining the validity period of the nonces r_i.

Round 2. On timeout, R broadcasts (R_s, R_s') to all tags in its range. Each tag_i in range of R that has not timed out, checks that $R_s' = h(K_s, R_s)$ and if so, computes:

$$M_i = h(K_s, r_i, ID_i),\ h(K_s, M_i) \oplus ID_i = M_i' \oplus ID_i,$$
$$P_i = h(K_i, r_i, R_s),\ P_i' = h(K_s, P_i),$$

sends $(r_i, M_i, M_i' \oplus ID_i, P_i, P_i')$ to R and timeouts. R computes $M_i' = h(K_s, M_i)$, retrieves ID_i and checks that $M_i = h(K_s, r_i, ID_i)$ and $P_i' = h(K_s, P_i)$. If these are correct, the reader verifies the integrity of the group by using the codewords ID_i. On timeout, if the list of identifiers is complete, it compiles the the grouping proof: $(T_s, r_1, \ldots, r_{n_g}, h(P_1, \ldots, P_{n_g}))$ and sends this to the verifier. If the list is not complete then the reader R uses RS decoding to recover the missing tag identifiers and informs the verifier. To

validate the proof, the verifier computes R_s, and the corresponding P_i's. Then, it checks that the received $h(P_1, ..., P_{n_g})$ is correct.

We shall assume that the keys K_g, K_i, K_s, the challenges T_s, R_s and the random numbers r_i, all have the same (bit) length κ, which is the *security parameter* of the protocol. The protocol has just two rounds and only requires tags to be able to generate random numbers and compute a hash function.

Protocol Integrity & Privacy

The proof is informal. Integrity is established by the use of authenticators, counters and timers. To forge a grouping proof the adversary must compute the MAC $h(P_1, \ldots, P_n)$.

An adversary that physically tracks a group of tags G can determine which executions are linked to this group; this cannot be prevented. Similarly an adversarial reader that is authorized to inspect G can link the inspected tags. Unlinkability concerns periods during which physical tracking or authorized inspection is interrupted. Since with each new session every tag_i updates its counter T_{s_i} and draws a fresh pseudo-random number r_i after receiving the reader's authorized challenge, the responses of the tags are pseudorandom and cannot be distinguished with probability better than negligible.

Common RFID Attacks

Replay attacks. The use of the counter T_s by the reader and the tags in the authenticators T_s' and r_i' prevents replay attacks: if an adversarial reader re-uses T_s, the tags that received this earlier will have updated their counter and will not respond. If a previous T_s was never sent to the tags, then the tags will respond (only this time) and a proof will be generated but this will not be accepted by the verifier (T_s is not valid). Similarly a replayed response (r_i, r_i') for a previous counter value T_s will not be valid.

Impersonation attacks on tags are prevented by using private keys K_i. Impersonation attacks on a reader will not yield a valid proof: only readers that have access to the one-time authorization (T_s, K_s) of the verifier can interrogate G. The authorizations P_i from different sessions cannot be used to compose a proof since R_s, which involves all the session nonces from the different tags and the counter of the verifier, is used to compute P_i.

De-synchronization attacks. If a protocol execution completes successfully then all tags will share the same counter value. No tag will accept a previously used T_s. Even if tags do not share the same counter value (*e.g.*, because of an interrupted interrogation), there are no synchronization concerns.

3.5 Metro-Scale Logistics: RS vs LDPC

$RS(n,k)$ codes are costly when n is large: encoding and decoding have quadratic complexity. However for pallet group scanning, the number of RFID tags is not large (typically not more than 100 tags), and the computational complexity is born by RFID readers for which the computational and memory constraints are not so strict, as opposed to the RFID tags that are severely constrained. Indeed the tag memory-erasure tradeoff for passive RFID tags is the limiting factor: for collections of $n_t = 100$ tags, with up to $s_t = 60$ missing tags, we need: roughly 144 redundancy + 20 bits (Burmester et al., 2016), which is within the bounds of EPC Gen2 (EPC-Global, 2015).

On the other hand Gallager LDPC codes can be decoded in linear time. However this benefit can only be realized by exceeding the memory constraints of passive RFID tags.

4 SMART GRIDS

4.1 Background

With the advancements in computer and communication technologies, analog controls are been replaced by networked digital devices that are far more effective to address the needs of smart structures because of their programmability, flexibility and efficiency. Historically, infrastructures relied mostly on proprietary technologies and were realized as stand-alone systems with analog devices in physically protected locations. The situation has changed considerably in recent years. Commodity hardware, software, and communication technologies are currently employed by host infrastructures to enhance their connectivity and improve the overall efficiency and robustness of their operations. Unfortunately this has also significantly increased their vulnerability to cyber threats, and hinder progress for more efficient management.

Former Defense Secretary Leon E. Panetta warned recently that the United States was increasingly vulnerable to foreign computer hackers who have gained access to the computer control systems of the nation's power grid, transportation system, financial networks and government ..., and we are facing the possibility of a "cyber-Pearl Harbor".[1] The Stuxnet attack (Langer, 2011) and other more recent malware

[1] http://blogs.wsj.com/cio/2012/10/11/u-s-defense-chief-warns-of-digital-911/

attacks such as Havex[2] and the 2015 power plant sabotage in Ukraine[3] show that the threats are very real.

4.2 Real-Time Availability for Grids

Electric Grids are mixed latency and throughput sensitive infrastructures. To secure them it is essential that the communication channels be protected in real-time: correct messages delivered at the wrong time could lead to erroneous system responses and failure. Communication must therefore be subject to stringent real-time requirements that render commonly adopted security paradigms for cyber-only systems inapplicable. For example, confidentiality, integrity and availability are the primary goals of cyber security with confidentiality (privacy) being most important. However for grids that are latency sensitive, availability and integrity become the primary goals, with confidentiality often a secondary goal.

There have been several efforts to secure electric grids primarily based on extending mechanisms already used to protect their separate cyber and physical components. However, there is no formal security framework that deals with software threats, hardware threats, network threats, and physical threats in a comprehensive way. Without a unified security framework, approaches to securing grids are ad-hoc and cannot provide proven requirements for real-world systems, therefore hindering their adoptability.

4.3 IEC 61850 & IEC 62351

IEC 61850 is a standard of the International Electrotechnical Commission for power utility automation (IEC61850, 2007) that integrates protection, control, measurement and monitoring. The standard offers advanced object oriented semantics for information exchange in power utility applications, SCADA, system protection, substation automation, etc, and supports advanced communication with an integrated, fully managed Ethernet switch and Internet TCP/IP communication protocols. The IEC62351 (IEC62351, 2015) standard is a security extension for data communication.

4.4 An IEC 61850 Resilience Framework for Compliant Systems

The Framework is based on the NIST Technical Guide for Security Assessment & Testing (SP800-115, 2008) discussed in Section 2.4 and extends IEC 62351 to capture system resilience. This involves the:

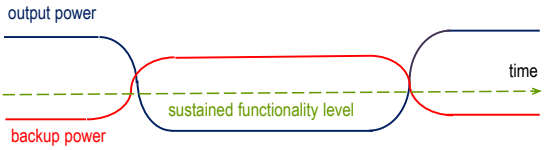

Figure 7: Maintaining functionality at a sustained level

1. Analysis of real-time multi-layer vulnerabilities & threats to electric grids resulting from untrusted or unexpected behavior.
2. Analysis of cascading electric grid faults and finding time-to-restore solutions that maintain sustained functionality levels.
3. Identification of vulnerabilities & threats of IEC 61850 compliant grids and the design policies/ mechanisms that support resilience.

The framework should be regarded only as a first step. Infrastructures are complex adaptive systems (Miller and Page, 2009), and protecting them has to be an ongoing process that evolves, and accounts for changes in policies/specifications and threat scenarios.[4] In particular, resilience must address:

- Adaptation threats: system policies and specifications must adapt to address emergent behavior. This may be driven by the adversary.
- Insider behavior and zerodays: Offensive security architectures and real-time stochastic analyzers.

4.5 Maintaining Functionality at Sustained Levels

Resilient systems must be self-healing: when a problem is detected in real-time, it must be isolated to maintain functionality at sustained power levels during emergencies, until the problem is addressed. Currently grids may experience cascading failures in emergency situations where outages are poorly contained.[5] Although the technologies for containing cascading failures continue to improve, their ability to contain failure in real-time implementations has not been tested (*e.g.,* hardware-in-the-loop testing). In particular in adversarial scenarios. A recent report

[2]https://ics-cert.us-cert.gov/alerts/ICS-ALERT-14-176

[3]http://www.theguardian.com/world/2015/nov/22/crimea-state-of-emergency-power-lines-attacked

[4]The Kerberos authentication protocol proposed in the late 1980's, was modified several times.

[5]NASEO: http://www.naseo.org/ data/sites/1/documents/publications/NASEO_Smart_Grid_and_Cyber_Security_for_Energy_Assurance_rev_November_2011.pdf

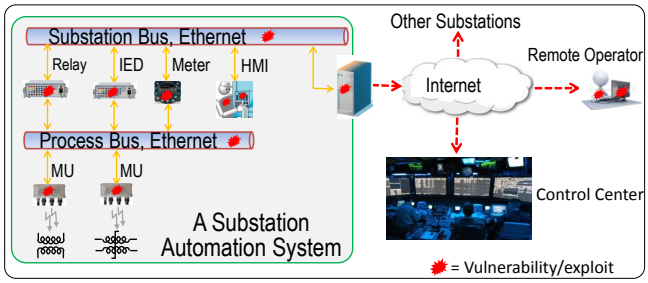

Figure 8: Vulnerabilities of an IEC 61850 enabled substation automation system.

in USA Today[6] observes that because electrical grids operate as an interdependent network failure in any one element requires energy to be drawn from other areas implying that if multiple parts fail at once this may lead to a cascading failure. The report mentions several game changer incidents such as the coordinated attack on the Metcalf substation of the Pacific Gas & Electric Co on 4/16/2013, as well as 362 other physical and cyber attacks on public utilities between 2011 and 2015.

Addressing cascading failures will require backup energy storage for time-to-restore capability (Figure 7), and significantly more automated controls. Several energy storage technologies can be used for backup power, such as pumped hydro, compressed air energy storage, flywheels, batteries and supercapacitors, hydrogen fuel cells, etc. Energy storage will have to be combined with a fast-simulation and modeling tool that gathers information, makes decisions and takes control actions.

4.6 Vulnerabilities of IEC 61850 Compliant Systems

Figure 8 shows the vulnerabilities of an IEC 61850 compliant substation automation system. In the "switchyard" (bottom left), Brick Merging Units (MU) provide diagnostics to monitor and protect power generators. In the control house (top left), Intelligent Electronic Devices (IED) and Relays monitor and control the power supply. The MU and IED are networked. One (or more) of the IED has Ethernet connectivity to a SCADA control system and a Human Machine Interface (HMI). Internet connectivity is available for software updates and other checks.

To analyze the vulnerabilities of the individual components of such systems one has to assess and test specific industrial realizations. The General Electric (GE) Multilin HardFiber System is an IEC 61850 compliant substation automation system that maps switchyard measurements to protection relays located in the control house using secure communications (a single fiber optic cable is used, eliminating most of the field wiring). This system consists of several components, including (Figure 9): a Multilin Brick Merging Unit (MU), Cross Connect Panels, the Protection Relays: C60 (Breaker Protection), F60 (Feeder Protection), the IEDs: G60 (Generator Protection), D60 (Transmission Line Protection), and N60 (Network Stability and Synchrophasor Measurement System). The transmission channels are either IEC61850 control fiber, switchyard fiber or TCP/IP channels (dotted red arrows). IEC61850 channels include: MMS (Manufacturing Message Specification), GOOSE (Generic Object Oriented Substation Events), and SMV (Sampled Measured Values) channels. Multilin uses: MultiLink ML2400 and MultiLink ML1600 switches for Ethernet and UR Switch Modules.

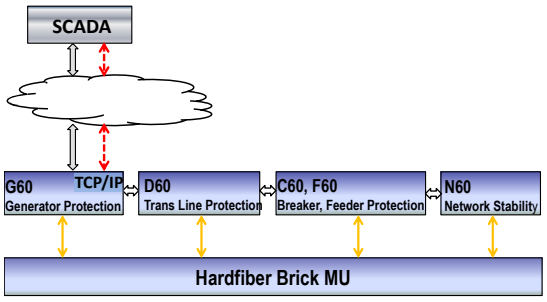

Figure 9: GE Multilin HardFiber System Architecture.

4.7 Threat Analysis

The IED G60 in Figure 9 has advanced automation capabilities for customized protection and control solutions, in particular for updating firmware via TCP/IP channels. This exposes the Multilin system to several threats in which an attacker can gain access to G60 via a TCP/IP channel (an *H2C* exploit,

[6]Steve Reilly, Special Report, 03.25.2015: http://www.13newsnow.com/story/news/2015/03/25/bracing-for-a-big-power-grid-attack-one-is-too-many/70417150/

Section 2.2) to deliver a malicious outcome, *e.g.,* to upgrade the system with compromised firmware, or for firmware manipulation/tampering.

G60 has an LED display panel that will indicate anomalies. However an attacker can manipulate the display to show an incorrect state of the network environment. G60 has also the ability to program user buttons. The attacker can reprogram one of the buttons (an $H2P$ exploit) and force a false LED alarm display. A controller may then push the button ($H2P \Rightarrow P2C$ attack) which in turn may disable a switch or shutdown a generator (a $C2P$ exploit). These exploits involve social engineering, or use vulnerabilities not previously identified.

The IED D60 and N60 use the IEEE 802.1q protocol to carry VLAN (Virtual Local Area Network) GOOSE traffic over the Ethernet. This protocol supports priority queueing. If an attacker $\mathcal{A}$ has access to the VLAN used by the IED to communicate (an $H2C$ exploit), then $\mathcal{A}$ can send high priority packets to the IEDs and prevent true high priority control packets from being processed. This could cause critical GOOSE messages to be delayed from processing, that may turn-off a breaker, due to invariances in the electric monitoring (a $C2P$ exploit).

Another vulnerability of the Multilin IEDs involves MMS IEC 61850 channels. If an attacker $\mathcal{A}$ can extract an SSH public key (*e.g.*, using an $H2C$ exploit) then $\mathcal{A}$ can use an MMS to update an IED via SSH, or install malware. If the IED do not support authorization and integrity then the IED can easily be compromised with LAN sniffing or by using device vulnerabilities (an $H2C$ exploit). For example, $\mathcal{A}$ can send a control messages to an IED to cause an electrical switch to open causing unknown issues to the electrical grid.

It is essential to stress test all IEC 61850 devices and determine their threshold behavior—the potential exploits. Also, to test the system for vulnerabilities that an attacker can exploit to gain access to the IED, Relays and MU, via the SCADA/HMI system.

4.8 Hardware-In-the-Loop (HIL) Testbeds

A number of testbeds for Industrial Control Systems (ICS) have been developed to discover new vulnerabilities and to analyze attack patterns and their impacts. Many of these support HIL testing. The most notable testbed is the National SCADA Test Bed (NSTB) that draws on the integrated expertise and capabilities of the Argonne, Idaho, Lawrence Berkeley, Los Alamos, Oak Ridge, Pacific Northwest, and Sandia National Laboratories to address the cybersecurity challenges of energy delivery systems.[7] The NSTB testbed combines real hardware (sensors, actuators, PLCs, IEDs, etc.), software (supervisory and control systems, etc.) and emulate HMI. There are also joint academia and industry testbeds such as the Trustworthy Cyber Infrastructure for Power Grids (TCIPG) of the University of Illinois, ExoGENI-WAMS-DETER of NC State University and ISI's DETER-lab. These are Real-Time-Digital-Simulator (RTDS) testbeds for HIL testing.

Analyzing memory corruption vulnerabilities of ICS embedded systems on hardware based testbeds often runs the risk of damaging or destroying the testbed hardware (Redwood et al., 2016) Since these testbeds are highly complex and costly, alternative solutions should be sought for memory corruption exploits. Software based ICS testbeds, unlike their hardware counterpart, are portable, and distributable. However virtualizing embedded systems is non-trivial, especially at the firmware level, with current solutions failing to distinguish certain attack patterns. Recent advances in physics simulation and microprocessor virtualization emulators have made it possible to consider methodologies that integrate simulated physics and embedded virtualization. This is a first step towards realistic software based ICS testbeds that can be used for onsite incidence response (Redwood et al., 2016).

4.9 Metro-Scale Grids: Soft vs Hard ICS Testbeds

As observed above, hardware based ICS testbeds are restricted to particular classes of malware analysis and cannot be used for onsite incidence response. Current software based testbeds can only emulate small networks. However recent advances in microprocessor virtualization emulators and physics simulation have made possible the design of soft ICS testbeds that can be used for onsite analysis without risk of damaging the testbed (Redwood et al., 2016).

5 CONCLUSIONS

We have considered two very different smart structure applications to illustrate the challenges of securing such structures. The first involves the supply chain: an untrusted RFID reader has to identify missing tagged objects in a pallet, and compile a proof of integrity if no tagged objects are missing. The reader

[7] http://energy.gov/oe/technology-development/energy-delivery-systems-cybersecurity/national-scada-test-bed

does not share any secret information with the tags, and is only given a one-time authenticator. The reader should not be able trace tagged objects of pallets that where not inspected (privacy). For applications with pallets having no more than 100 tagged items this is possible by using RS erasure codes. Larger pallets require a different approach, since for these the memory-erasure tradeoff is excessive for passive RFID tags.

The second application involves IEC 61850 compliant industrial systems: we considered the GE Multilin HardFiber system. In this case resiliency can only be established if the components of the system function as intended. In particular, they should not be compromised. To establish this, ICS testbeds are used. Hardware testbeds are costly and risk being damaged if the memory of tested components is corrupted. Also, they cannot be used for onsite malware analysis. On the other hand software testbeds can only emulate small networks. For high-end metro-scale applications a software based approach is proposed (a proof of concept) that integrates simulated physics and embedded virtualization.

REFERENCES

Abadi, M., Budiu, M., Erlingsson, Ú., and Ligatti, J. (2009). Control-flow integrity principles, implementations, and applications. *ACM Transactions on Information and System Security (TISSEC)*, 13(1):4.

Redwood, W.O., Reynolds, J., and Burmester, M. (2016). Soft ICS Testbeds: A Simulated Physics and Embedded Virtualization Integration (SPAEVI) Methodology. In Rice, M. and Shenoi, S. editors, *Critical Infrastructure protection X*, Springer.

Burmester, M. and Munilla, J. (2016). An Anonymous RFID Grouping-Proof with Missing Tag Identification. *10th IEEE International Conference on Radio-Frequency Identification,* 3-5 May, Orlando. U.S.A.

Beaver, D. (1989). Multiparty protocols tolerating half faulty processors. In Brassard, G., editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 560–572. Springer.

Ben Mabrouk, N. and Couderc, P. (2015). EraRFID: Reliable RFID systems using erasure coding. In *RFID, 2015 IEEE International Conference*, pages 121–128.

Burmester, M., de Medeiros, B., and Motta, R. (2008). Provably Secure Grouping-Proofs for RFID Tags. In Grimaud, G. and Standaert, F.-X., editors, *CARDIS*, volume 5189 of *Lecture Notes in Computer Science*, pages 176–190. Springer.

Burmester, M., Magkos, E., and Chrissikopoulos, V. (2012). Modeling Security in Cyber-Physical Systems. *International Journal of Critical Infrastructure Protection (IJCIP)*, 5(3-4):118–126.

Burmester, M. and Munilla, J. (2013). *Security and Trends in Wireless Identification and Sensing Platform Tags: Advancements in RFID*, chapter RFID Grouping-Proofs. IGI Global.

Canetti, R. (2001). Universally composable security: a new paradigm for cryptographic protocols. *Proceedings, 42nd IEEE Symposium on In Foundations of Computer Science, Foundations of Computer Science*, pages 136–145.

Chien, H.-Y., Yang, C.-C., Wu, T.-C., and Lee, C.-F. (2009). Two rfid-based solutions to enhance inpatient medication safety. *Journal of Medical Systems*.

EPC-Global (April, 2015). Radio-Frequency Identity Protocols, Generation-2.V2. UHF RFID. Technical report.

Guidry, D., Burmester, M., Yuan, X., Liu, X., Jenkins, J., and Easton, S. (2012). Techniques for securing substation automation systems. In *7th Int. Workshop on Crit. Inform. Infrastr. Secur.(CRITIS)*.

Huang, H.-H. and Ku, C.-Y. (2008). A rfid grouping proof protocol for medication safety of inpatient. *Journal of Medical Systems*.

ICS-CERT (2015). *Cyber Threat Source Descriptions*. Industrial Control Systems, Cyber Emergency Response Team.

IEC61850 (2007). IEC 61850 Parts 1-10, Power Utility Automation. *http://www.iec.ch/smartgrid/standards/*.

IEC62351 (2015). IEC 62351 Parts 1-8, Information Security for Power System Control Operations. *http://www.iec.ch/smartgrid/standards/*.

Juels, A. (2004). "Yoking-proofs" for RFID tags. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pages 138–142, Washington, DC, USA. IEEE Computer Society.

Juels, A. (2006). Generalized "yoking-proofs" for a group of RFID tags. In *MOBIQUITOUS 2006*.

Kapoor, G. and Piramuthu, S. (2012). Single RFID Tag Ownership Transfer Protocols. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 42(2):164–173.

Kocher, P. C. (1989). Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Advances in Cryptology, CRYPTO '89, 9th Annual International Cryptology, Proceedings*, volume 1809 of *Lecture Notes in Computer Science*, pages 104–113. Springer.

Langer, R. (2011). Cracking Stuxnet, a 21st-century cyber weapon. *Entertainment and Design*.

Liu, H., Ning, H., Zhang, Y., He, D., Xiong, Q., and Yang, L. T. (2013). Grouping-proofs-based authentication protocol for distributed rfid systems. *IEEE Trans. Parallel Distrib. Syst.*, 24(7):1321–1330.

Miller, J. H. and Page, S. E. (2009). *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*. Princeton University Press.

Munilla, J., Guo, F., and Susilo, W. (2013). Cryptanalysis of an EPCC1G2 Standard Compliant Ownership Transfer Protocol. *Wireless Pers Commun*, (72):245–258.

Piramuthu, S. (2006). On existence proofs for multiple RFID tags. In *IEEE International Conference on Pervasive Services, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU 2006*, Lyon, France. IEEE, IEEE Computer Society Press.

RAE (2012). *Smart infrastructure: the future, The Royal Academy of Engineering*. ISBN 1-903496-79-9.

RFC6816 (2013). Simple Low-Density Parity Check (LDPC) Staircase Forward Error Correction (FEC) Scheme for FECFRAME.

RFC6865 (2013). Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME.

Richard G. Little (2002). Controlling Cascading Failure: Understanding the Vulnerabilities of Interconnected Infrastructures. *Journal of Urban Technology*, pages 9:109–123.

Roemer, R., Buchanan, E., Shacham, H., and Savage, S. (2012). Return-oriented programming: Systems, languages, and applications. *ACM Transactions on Information and System Security (TISSEC)*, 15(1):2.

Saito, J. and Sakurai, K. (2005). Grouping proof for RFID tags. In *19th International Conference on Advanced Information Networking and Applications, AINA 2005.*, volume 2, pages 621–624.

Sato, Y., Igarashi, Y., Mitsugi, J., Nakamura, O., and Murai, J. (2012). Identification of missing objects with group coding of RF tags. In *RFID, 2012 IEEE International Conference on*, pages 95–101.

SP800-115 (2008). NIST, Technical Guide to Information Security Testing and Assessment.

Standaert, F.-X., Malkin, T. G., and Yung, M. (2009). A unified framework for the analysis of side-channel key recovery attacks. In *Advances in Cryptology-EUROCRYPT 2009*, pages 443–461. Springer.

Su, Y. and Wang, C. (2015). Design and analysis of unequal missing protection for the grouping of rfid tags. *Communications, IEEE Transactions on*, PP(99):1–1.

Su, Y.-S. (2014). Extended Grouping of RFID Tags Based on Resolvable Transversal Designs. *Signal Processing Letters, IEEE*, 21(4):488–492.

Su, Y.-S., Lin, J.-R., and Tonguz, O. K. (2013). Grouping of RFID Tags via Strongly Selective Families. *IEEE Communications Letters*, 17(6):1120 – 1123.

Su, Y.-S. and Tonguz, O. K. (2013). Using the Chinese Remainder Theorem for the Grouping of RFID Tags. *Communications, IEEE Transactions on*, 61(11): 4741–4753.

The White House (2013). *Executive Order, Improving Critical Infrastructure Cybersecurity*. Office of the Press Secretary.

Yampolskiy, M., Horvath, P., Koutsoukos, X. D., Xue, Y., and Sztipanovits, J. (2013). Taxonomy for description of cross-domain attacks on cps. In *Proceedings, 2nd ACM International Conference on High Confidence Networked Systems*, pages 135–142. ACM.