Cloudifying Critical Applications: a Use Case from the Power Grid Domain

F. Campanile², L. Coppolino^{1,2}, S. D'Antonio^{1,2}, L. Lev³, G. Mazzeo¹, L. Romano^{1,2}, L. Sgaglione¹, F. Tessitore¹

¹University of Naples "Parthenope", IT

{luigi.coppolino, salvatore.dantonio, giovanni.mazzeo, luigi.romano, luigi.sgaglione, francesco.tessitore}
@uniparthenope.it

²Sync Lab s.r.l., IT, f.campanile@synclab.it

³Israel Electric Corporation, IL, leonid.lev@iec.co.il

Abstract—The cloud computing paradigm is gaining more and more momentum, to the extent that it is no more confined to its initial application domains, i.e. use by enterprises and businesses that are simply willing to lower costs or to increase computing capacity in a flexible manner. In particular, increasing interest is recently being paid to the dramatic potentials that the use of cloud computing technology by critical infrastructure (CI) operators might bring about, in terms of benefits for the society at large. Since accidental or deliberate damage to a CI may result in devastating consequences, this mandates for dependable and trustworthy security mechanisms in cloud platforms. In this paper, we present a distributed application for real-time monitoring of a Power Grid. The application, which is called PoGriMon, is deployed on top of the *SecureCloud* platform, a security-enhanced IaaS solution that exploits the Intel Software Guard eXtension (SGX) technology. PoGriMon has been designed based on the requirements of the SCADA network of the Israeli Electric Corporation (IEC), and it is currently being validated in a realistic setup also provided by IEC.

Keywords—power grids; SCADA; critical infrastructures; cloud technology; Intel Software Guard eXtension (SGX), real-time monitoring.

I. INTRODUCTION

The industrial community is now becoming aware of the tremendous advantages that cloud technology may bring about. The cloud computing paradigm is a promising solution in many fields, due to its high value in terms of cost savings. However, in application domains – such as CIs – where security is an issue, companies are reluctant to move to the cloud, also due to the dramatic increase – both in number and complexity – of attacks experienced in the last few years. CIs have more stringent security requirements than possibly any other sector, due to the terrible impact that attacks directed to them may have on the safety, prosperity, and well-being of the society itself. Evidence such as the "Black Energy 3" attack enforced in Ukraine [1] in 2015 or the "Havex" SCADA-focused trojan launched in 2014 [2], demonstrates how important the security of Industrial Control Systems (ICS) is. In order for cloud technology to really take up in the CI domain, effective solutions are needed to meet the challenging security requirements of applications. In this work we present PoGriMon (PGM), a distributed cloud-based

application for real-time monitoring of a power grid. We describe the conceptual architecture of the application, discuss the implementation details of the current implementation, and illustrate how it uses the advanced security features provided by the underlying cloud platform. The cloud platform supporting PoGriMon is called *SecureCloud*, and is being developed within the context of the Secure Big Data Processing in Untrusted Clouds (*SecureCloud*) project¹. To enable PoGriMon to protect the confidentiality and integrity of the data that it gathers from power grid SCADA sensors, the *SecureCloud* platform exploits Software Guard eXtension (SGX), the new extension of Intel Instruction-Set Architecture (ISA) provided in the newest Skylake processors. SGX permits to protect application state from the hypervisor and even from the operating system. Since all data is encrypted in memory in Secure Enclaves, and only the CPU has access to the encryption keys, even privileged access to a machine (i.e. the cloud provider itself) does not help to gain access to data protected with SGX. The incorporation of secure enclaves in the cloud computing software stack results in superior security guarantees to applications – such as PoGriMon – which handle sensitive data.

The *SecureCloud* platform provides integrity and confidentiality of sensitive data to the PoGriMon application through: 1) SGX-enabled Secure Containers (SCONE) (Arnautov et al. [15]) able to protect the processing of sensitive code and data against powerful attackers; 2) an SGX-enabled event bus (namely: ZMQ) able to protect messages exchanged among containers by encrypting them inside enclaves.

To enable PoGriMon to exploit the enhanced security features provided by the *SecureCloud* platform, the application is split in a set of micro-services, which are executed in distinct secure containers. The resulting architecture, therefore, consists of multiple secure containers exchanging messages through the secured ZMQ event bus.

The remainder of this paper is organized as follows. Section **Error! Reference source not found.** provides an overview of related research initiatives. Section II analyzes the type of threats that may arise in a cloud environment. Section III describes the use case, and specifies its security requirements. Section IV provides insight into Intel SGX, by describing limitations of this innovative technology and how such limitations have been

¹ <https://www.securecloudproject.eu/>

addressed. Section V gives an overview of the *SecureCloud* platform and of its main architectural elements. Then, section VI describes the PoGriMon application and its integration with the *SecureCloud* platform. Finally, section VII concludes the document.

II. THREATS TO CIs IN A CLOUD ENVIRONMENT

Although the migration of CI applications to the cloud lays the foundations for better quality services – characterized by a higher resistance to trivial attacks and, most of all, by a substantial improvement in terms of reliability – a number of new security threats arises. Data loss or compromise, loss of organizational control, account hijacking and Denial of Service (DoS) are the noteworthy risks that organizations must take into account if moving to a cloud based IT infrastructure. The research community is proposing new security solutions, as testified in [5], where authors survey current available security mechanisms in cloud infrastructures to address most relevant open issues.

The adaptation of cloud-hosted SCADA systems to the cloud migration is paramount, due to the lack of appropriate security mechanisms in the most common SCADA protocols (such as Modbus and DNP3), which do not support or perform authentication, nor encryption.

According to NIST, cloud computing "presents certain unique security challenges resulting from the cloud's very high degree of outsourcing, dependence on networks, sharing (multi-tenancy), and scale" [4]. Fernandes et al. [8] provides a survey of research literature to highlight cloud security open issues and challenges. Some of them specifically affect Industrial Control Systems (ICS). We report here the most relevant ones:

- T1: Data Breach - CIs data is the main target of malicious intrusive actions. Data confidentiality is certainly important, but it is our belief that data integrity is much more important because attacks (e.g., manipulating sensor or control data) on the SCADA system could have terrible effects on people safety, as it (mis)leads operators to take wrong decisions. Just think of the impact that false information on the status of the power grid may have on a distribution network. The problem of data outsourcing in a cloud migration is thus of immediate concern.
- T2: Account or Service Traffic Hijacking - The attempt to impersonate legitimate operators is a possible way through which hackers can do malicious actions against the CI. The intruder can get into critical areas of a distributed monitoring service and possibly compromise the confidentiality, integrity, and availability of those services.
- T3: Denial of Service (DoS) - The high availability of SCADA systems is a fundamental requirement. Precisely for this reason, attackers may aim at an outage of CI services through a DoS or Distributed DoS (DDoS) attack. These attacks may even be more dangerous in a cloud environment, since when the workload increases with respect to a specific service, the cloud environment provides additional

computational power to that service. This means that on the one hand the cloud system counters the effects of the attack, but on the other hand it supports the attacker in his/her evil activity, by providing him/her with more and more resources.

- T4: Shared Technologies Vulnerabilities - The multi-tenancy feature of cloud technologies is extremely risky in terms of security if the hypervisor is not properly secured. An evil activity on CI data integrity and confidentiality may be launched through penetrations in virtual machines (VM) residing on the same hypervisor.
- T5: Malicious Insiders - Malicious administrators of the Cloud Provider (CP) or any other system user with privileged access to resources are a consistent threat that traditional security mechanisms are hardly able to cope with.

III. REQUIREMENTS OF THE IEC USE CASE

A civil electric supply and distribution network infrastructure has been chosen as a use case to validate and demonstrate our work. Such an infrastructure is extremely representative, since it is composed by assets like transmission lines and distribution substations that make it a safety-critical infrastructure with challenging security requirements.

Our use case is under the administration of a public and government-owned company, namely the *Israel Electric Corporation (IEC)*. Activities include the generation, transmission and transformation, distribution, supply, and sale of electricity. The company supplies reliable and high-quality electricity, complies with leading service standards, maintaining economic, commercial and environmental principles.

A cloud-based real-time monitoring application is an attractive solution for IEC administrators because they can get rid of the burden (and reduce the costs) of IT systems management and easily interconnect the assets, which are geographically distributed. Also importantly, cloud technology seems to be an attractive option based on the requirements (functional and non-functional) that are defined in detail in the following subsections.

A. Functional Requirements

The following main requirements with respect to cyber security must be observed for this application. Data exchange between legitimate users and the application and data stored in the cloud must not be accessible by unauthorised parties, nor it should be possible to alter them undetectably. Additionally, the business logic of the processes should not be altered. It is thus necessary - among other things - to establish a secure communication mechanism for connecting the various processes. More specifically, the functionalities needed by the application are those typical of a SCADA system, that is to say:

- FR1 - Sensor Data Acquisition - data acquisition is needed from sensors, meters and field devices, such as photo, pressure, temperature, and flow sensors that

monitor fundamental parameters for service quality and infrastructure integrity.

- FR2 - Sensor Data Collection, storage and retrieval - The enormous amount of data acquired by the sensors must be handled and stored for subsequent analysis such as statistical trends, regulations, future load planning, and billing.
- FR3 - Event and Alarm Processing - Any suspicious activity should be reported by the monitoring application. An Alarm Management System (AMS) must clearly show appropriate alarms indicating abnormal situations, rank priorities and warn about consequences. An history of the alarms must be securely stored. Reactions must be proposed for alarms of different level of importance in case sensors measurements go out of pre-defined thresholds. A correlation of events of different nature should be done to spot patterns of complex attacks.
- FR4 - Monitoring through Human Machine Interface (HMI) - Operators need to have an intuitive interface to the system, to be informed of the status of the infrastructure in a timely fashion, to be able to react promptly to alerts/alarms. They also need to have access to the historical data, to make plans for improvements.

B. Para-Functional Requirements

Power grids are unarguably exposed to risks that are related to natural phenomena, disasters, and criminal/terrorist activity. For this reason, the non-functional requirements imposed, especially those related to security and dependability, are particularly stringent, which is the main cause of skepticism with respect to migration of CI applications to the cloud. The monitoring application should satisfy the following non-functional requirements:

- NFR1 - Data Security - Critical decisions are usually taken based on analysis of the measurement data provided by the monitoring system. This means that data confidentiality - and even more data integrity - are of primary importance. The monitoring application should enforce mechanisms able to protect data from malicious accesses and modifications.
- NFR2 - Service Availability and Responsiveness - A timely response to problems is an important requirement, e.g., to have operators quickly react to critical conditions such as a terrorist attacks to the infrastructure or the monitoring system. It is also equally important to guarantee continuous service provision, since an outage could be fatal for the safety of the population.
- NFR3 - Service isolation - In a CI, any asset associated with the monitoring system must comply with a number of regulations and standards. Validation of compliance is a complex and expensive process, especially when third-party cloud providers are involved.
- NFR4 - Scalability - The monitoring application must be able to cope with load peaks. This capability must be

maintained also when the number of data feeds grows (i.e. more sensors or event analysers are added to the infrastructure).

- NFR5 - Interoperability - The application must support seamless integration of technologies of different nature, and also allow the usage of software modules written in different programming languages.

IV. FUNDAMENTALS OF INTEL SGX

Intel SGX is the new ISA extension of Intel's CPUs that allows to protect sensitive code and data even from privileged instructions executed from *Ring0*. SGX can be conceived as a "Reverse Sandbox". The threat model, in fact, starts from the assumption that all the external world – including system software in general and the OS in particular – is untrusted. Hence, integrity and confidentiality of sensitive code and data must be protected even against those platform components. SGX does so by creating a reserved memory space known as *enclave* able to protect an application from the external environment (including the OS or the Hypervisor). Access to the enclave memory area is protected by security check mechanisms enforced by the CPU. The enclave contains the sensitive data and the code that uses it. SGX, therefore, enables users to enforce the security of their code and data without having to trust the cloud provider.

Key point of SGX is the *software attestation* [11] feature, useful to prove the goodness of a piece of software running in an enclave. Such a process convinces an enclave that it is communicating with another enclave that has a specific measurement hash, that it is running in a secure environment and has not been modified. The mutual verification between the enclaves is enforced by using a processor key, which is accessible only by a special enclave known as *Quoting Enclave*.

SGX supports both an intra-attestation and an inter-attestation service, that is the SGX-enabled procedure of enclave identity verification can be empowered between two enclaves residing on the same host or between two enclaves residing in different hosts. The remote attestation service builds a secure channel between the two enclaves by performing a Diffie-Hellman key exchange.

The high level of security provided by SGX does not come without a cost. In fact, there are some limitations that make the development of SGX-based application a non-trivial engineering task. First, the security checks enforced by the CPU when an application wants to access an enclave have a non-negligible impact on performance. Therefore, the developer needs to reduce as much as possible the amount of EENTER and EEXIT SGX calls. Second, system calls cannot be performed within an enclave, since the OS is considered untrusted in the SGX threat model. This obviously imposes restrictions on what the application can and cannot do in enclaves.

In the following section, we provide more details about how the *SecureCloud* platform, on top of which the PoGriMon application runs, addresses such issues.

V. THE SECURECLOUD PLATFORM SUPPORTING POGRIMON

In this section we provide an insight into the *SecureCloud* platform describing the main architectural elements that help to ensure the confidentiality and integrity of data managed by the PoGriMon application.

The *SecureCloud* platform makes use of containers to execute the PoGriMon application in the cloud. Containers are a convenient solution, which is gaining more and more momentum. Compared to Virtual Machines (VMs), they have lower memory overhead, faster startup, and in general they use less resources. However, containers provide lower security than full-fledged VMs or dedicated servers, as they cannot provide the same level of isolation. The *SecureCloud* platform combines the efficiency of containers with the security of Intel SGX enclaves, in a mechanism called Secure Container. Secure Containers provide a protected execution environment where micro-services can run. This is a convenient paradigm for distributed cloud based application with challenging scalability and availability requirements.

Figure 1 reports the high level architecture on which PoGriMon executes. SGX-enabled containers running on the cloud execute the different micro-services belonging to PoGriMon. Each micro-service resides in a secure container that protects its sensitive parts in a SGX enclave. The untrusted part of a container, outside of the enclave, executes the micro-service framework runtime system, which is responsible for the management of the enclave and provides access to operating system functionalities (such as sending data over the network or storing files).

Figure 1 High-level architecture of *SecureCloud*

Three issues have been addressed, which are extremely relevant in terms of security:

- *Secure data processing within containers* – the *SecureCloud* platform leverages the SCONe secure container mechanism described in [15]. The architecture of SCONe provides transparent encryption and authentication of data through shields. The sensitive part of PoGriMon micro-services executes in the enclaves and communicates with the outside through the shielded interface. Moreover, thanks to asynchronous system calls and to the M:N correspondence of SGX enclave threads to OS threads, SCONe containers are able to

address one of the main issues of SGX, that is the degradation of performance.

- *Secure data communication between containers* – As already mentioned, the application consists of a set of micro-services connected via an event bus (a secured version of ZMQ). All messages on the event bus are encrypted. Encryption and decryption are performed automatically within enclaves. Decryption keys are stored securely within enclaves only. Authentication is also provided, as well as additional security properties of the event bus, such as ensuring freshness of the delivered messages, which is guaranteed by the event bus stubs that run in an enclave. For example, the event bus stubs detect duplicate messages, thus preventing replay attacks.
- *Secure data communication to the outside* – To enable micro-services to communicate with external clients and services, we mainly rely on the widely-used REST communication pattern via HTTP over SSL/TLS (HTTPS). The mapping between HTTPS and the event bus is achieved via a bridge: we terminate the TLS connection in an enclave and then distribute data across *SecureCloud* micro-services using encrypted channels, as illustrated in Figure 2. Typically, TLS connections terminate in proxies that distribute the requests via HTTP within a data centre. In the *SecureCloud* platform TLS connections are terminated transparently within a REST bridge. Requests/messages are then sent within a data centre via the encrypted event bus.

Figure 2 *SecureCloud* bridge (proxy) mapping HTTPS connections to secure channels

VI. THE POGRIMON CLOUD-BASED APPLICATION

In this section, we describe the architecture of the PoGriMon application. We assume that the cloud modules of the application run on a *Metal-as-a-Service (MaaS)* Cloud infrastructure equipped with SGX-enabled processors. We also assume that the on-site machine is SGX-enabled (in order to provide confidentiality of the collected data). First we discuss how data flows from the PoGriMon application to the cloud (and vice versa). Then we describe the integration of PoGriMon with the *SecureCloud* architecture. Finally, we motivate the main technological decisions that we have taken to implement the current version of the application.

A. PoGriMon Data Flow

Figure 3 provides a high-level view of the architecture of the PoGriMon application. On the infrastructure side, sensors are deployed on the field and are responsible for retrieving

measurements related to specific physical phenomena that are to be monitored. Measurement signals are then converted to digital data by means of signal converting devices, known as Remote Terminal Units (RTUs).

The data acquired by the sensors is then transmitted to a *Data Communication Layer (DCL)*. This consists of "Data Logger" equipment, which allows secure communication between the RTUs and the control/monitoring system by means of heterogeneous data communication layers, relying on different physical communication media and technologies, both wired and wireless. Multiple devices are deployed, each one responsible of a group of sensors installed nearby.

Sensor data collected by the DCLs is then handed over to *Gateway* machines in charge of preprocessing and then transmitting to other application components (running on the cloud platform) for further processing. A gateway acts as a bridge between the "edge" components of the monitoring application and the cloud, by implementing interfaces for decoupling communication protocols (the ones used by the monitored infrastructure and the ones used on the cloud side). Therefore, the Gateway interfaces on the one side with the ModBus protocol to acquire data from the DCL and on the other side with the *SecureCloud* event bus.

Therefore, the data gathered by the on-site application modules is sent to the *Message Broker Layer* (the event bus), which then distribute data, messages, and events to all the units involved, in charge of specific functions (e.g. data storage management, data computation, data provision, and alarm management). Storage of critical data, as well as batch and real-time processing facilities are provided by the cloud platform. Except for the data acquisition modules, all application components run on top of the cloud platform.

Figure 3 PoGriMon components and interfaces

B. Design of PoGriMon

Starting from the requirements imposed by the PoGriMon use case and from the need of a well-partitioned application to execute into SecureCloud Secure Containers, we revised the architecture of typical SCADA systems to finally create a secure reactive micro-service based cloud application. Reactive applications are characterized by asynchronous processing and are widely used in the Internet of Things (IoT) world, they are typically highly interactive, scalable, resilient, responsive and event driven; a set of characteristics perfectly in line with the application Non-Functional requirements defined in III.B.

Hence, we partitioned PoGriMon in multiple micro-services exchanging messages between them through *Publish/Subscribe* or *Point-to-Point* communication patterns. The idea is that each sensor data is published on the event bus by the *Gateway* and all the interested micro-services subscribe themselves to the topic (or the sensor) of interest.

The event bus chosen has been ZMQ which represented the best choice for our goals since it is lightweight and easy to extend for SGX enclaves interactions. The communication endpoints of a logical connection are inside SGX enclaves, where encryption and decryption take place. Data transmitted in the ZMQ channels is therefore always encrypted. The encryption keys are securely kept within the enclave. Their establishment is done through the TLS handshake protocol.

In the following, we report the micro-services that belongs to the PoGriMon application. For each of them we highlight the criticality in terms of security and where the confidentiality and integrity need to be protected. How this has been faced will be subject of next subsection.

- The *Data Collector* (or Data Publisher) acquires data from the sensors by communicating with a ModBus protocol with the DCL layer. The data is then transmitted to the ZMQ event bus responsible of delivering it to the other micro-services subscribers running at cloud level. Such micro-service needs to harden the data transmitted to the *SecureCloud* platform in order to make sure that the confidentiality and integrity won't be compromised.
- The *Access Controller* is responsible for the Identity Access Management (IAM) of PoGriMon in order to allow a policy-based access to the resources. The credentials of users reside in a Key-Value storage system and need to be protected into it.
- The *Alarm Manager* which enforces the stream processing to signal alarms or critical conditions that may arise in the IEC infrastructure. It does so through a Complex Event Processing (CEP) system that identifies and analyzes cause-and-effect relationships among events in real time. Every abnormal situation generates events interpreted by the CEP that, based on the rules defined by the administrator, enforces a correlation process and outputs a single alarm level. The event correlation of the CEP is extremely sensitive in terms of security: the integrity of events cannot be compromised.

- The *InMemory-Archiver* responsible for storing real-time data into a memory caching system (i.e. Memcached) for immediate processing. The data in RAM must be protected from malicious attacks.
- The *SQL-Archiver* responsible for the storing of historical data into a SQL database. All data in the database is encrypted at rest, and need to be accessible only to clients holding the decryption key.
- The *Web Proxy* which provides data coming from the application back-end to the front-end dashboard. This micro-service is responsible to protect the data flowing to the outside.
- The *Dashboard GUI* which provides a web-based Human Machine Interface (HMI) to monitor sensors' status. Such a GUI needs to establish a secure channel with the WebProxy in order to protect the final showing of measurements.

PoGriMon needs to protect the confidentiality and integrity of the data flowing through the previous micro-services. They need to be partitioned in order to properly execute their sensitive parts into Secure Containers of the *SecureCloud* platform and to encrypts sensitive data before sending to the ZMQ event bus. How the security is hardened will be subject of next subsection where the integration of PoGriMon with the *SecureCloud* platform is described.

C. Integration of PoGriMon with SecureCloud Platform

The PoGriMon application leverages the *SecureCloud* technologies to harden the sensitive part of its micro-services. A process of integration is therefore required in this sense.

A first phase carried out has been the identification of what need to be secured and what not. At a high level, the PoGriMon application must be hardened in: i) The provision of data exchanged between the *DataCollector* micro-service and the on-cloud application modules; i) The micro-services running on the Cloud. In respect to what described in section VII-B, the first means enforcing security between the *DataCollector* and the micro-services running on-cloud. The on-site gateway and the broker on the cloud platform need to attest each other's identities, establishing in this way a secure channel. Data is exchanged only between enclaves, and is never decrypted outside of them. This was accomplished by leveraging the *remote attestation* feature of Intel SGX: the idea is that the *DataCollector* micro-service collects the data, put more measurements in a single bucket to improve performances, and send the bucket to the stub into the enclave. At this point, the enclave performs the *attestation* of the enclave running on the *SecureCloud* platform and uses the key obtained with the Diffie-Hellman exchange mechanism to securely encrypt the bucket of sensitive measurements data into the enclave. The data is finally sent to a specific micro-service in charge of receiving such messages and publish them to the others.

The protection of the on-cloud micro-services is realized through the technologies of the *SecureCloud* platform: the micro-services are made up of code that executes partly outside an enclave (in the untrusted part of *SecureCloud* containers), and

partly inside the enclave (in the trusted part of *SecureCloud* containers). Hence, each micro-service listed in section VII-B runs into the untrusted part of a single container, when he needs to carry out a critical operation like a decryption of data received by someone else, it asks to the *shielded* interface exposed in SCONE containers to switch to the trusted part of the runtime system and perform the decryption. SCONE enforces a transparent encryption of files, of communication channels via TLS, and of console streams. Keys are stored into SGX enclaves, the micro-service can process encrypted data that is decrypted only inside the CPUs, and therefore inaccessible to the operating system. Each secure container uses a startup configuration file (SCF). The SCF contains keys to encrypt standard I/O streams, a hash of the FS protection file and its encryption key, application arguments and environment variables. Only an enclave whose identity has been verified can access the SCF.

In terms of communication among micro-services, as already said these make use of ZMQ. On top of it the *local attestation* of Intel SGX, privacy and integrity checks are performed through adequate key management and standard hashing, like SHA1, and symmetric and asymmetric encryption algorithms, like AES and RSA. The key management enforced by SGX is extremely powerful: the enclave generates a key pair into it, so inaccessible even from the kernel OS, seal the private data into the enclave, shares the public key with the authenticated clients, starts the communication with the client, which encrypt the data with the public key. Such data is decrypted only into the CPU. Besides, protection against attackers will be tackled by using the mechanisms provided by the Intel SGX enabled processors in all services which are supposed to run in third party infrastructure providers. The interface between clients and micro-services are established in a language-agnostic manner by using JSON data-interchange format. As a higher-level building block for both system and application communication infrastructure, we propose a secure content-based routing system. Content-based routing (CBR) is a flexible and powerful paradigm for scalable communication among distributed processes. It decouples data producers from consumers, and dynamically routes messages based on their content. Two main types of communication mechanisms are therefore used. First, point-to-point channels for secure communication across enclaves. Second, higher-level many-to-many communication primitives based on the publish/subscribe interaction model.

VII. CONCLUSIONS AND FUTURE WORK

In this paper we described the approach we have taken in the *SecureCloud* project to design a secure cloud application for a critical infrastructure. The platform uses the new Intel ISA extension known as SGX. We discussed the positive effects of a migration of critical infrastructure applications from traditional IT technology to the cloud, and analyzed the security challenges related to this technology shift. Then, we presented the architecture and the current implementation of PoGriMon, a cloud-based application for real-time monitoring of a real power grid, specifically the electric supply and distribution network of IEC. In the next months, we will perform a massive experimental campaign, aiming at validating the pilot

application. This will not be done on the real IEC infrastructure (for obvious reasons), but on an emulated environment that closely mimics the real setup.

ACKNOWLEDGEMENTS

SecureCloud has received funds from the European Union's Horizon 2020 research and innovation programme and was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under grant agreement No 690111.

The authors wish to explicitly thank the IEC technical staff (particularly: Radi Kalman and Michal Fisher) for their support in defining the requirements and setting up the experimental platform. They also wish to thank the *SecureCloud* partners who are in charge of developing the underlying cloud platform for the priceless technical discussions.

REFERENCES

- [1] Attack to ICS: Black Energy <https://ics-cert.us-cert.gov/alerts/ICS-ALERT-14-281-01B>
- [2] Attack to ICS: Havex <https://ics-cert.us-cert.gov/alerts/ICS-ALERT-14-176-02A>
- [3] S. Bera, S. Misra and J. J. P. C. Rodrigues, "Cloud Computing Applications for Smart Grid: A Survey," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1477-1494, May 1 2015. doi:10.1109/TPDS.2014.2321378K. Elissa, "Title of paper if known," unpublished.
- [4] Timothy Grance; Wayne Jansen; "Guidelines on Security and Privacy in PublicCloud Computing"; NIST
- [5] Luigi Coppolino, Salvatore D'Antonio, Giovanni Mazzeo, Luigi Romano, Cloud security: Emerging threats and current solutions, *Computers & Electrical Engineering*, ISSN 0045-7906, <http://dx.doi.org/10.1016/j.compeleceng.2016.03.004>. Keywords: Cloud computing security; Security techniques; Intel SGX; Homomorphic cryptography; Cloud platforms
- [6] T. Baker, M. Mackay, A. Shaheed and B. Aldawsari, "Security-Oriented Cloud Platform for SOA-Based SCADA"; *Cluster, Cloud and Grid Computing (CCGrid)*, 2015 15th IEEE/ACM International Symposium on, Shenzhen, 2015, pp. 961-970. doi: 10.1109/CCGrid.2015.37
- [7] Coppolino, L., D'Antonio, S., Formicola, V., Romano, L. Integration of a system for critical infrastructure protection with the OSSIM SIEM platform: A dam case study (2011) *Lecture Notes in Computer Science*, 6894 LNCS, pp. 199-212.
- [8] Diogo A. Fernandes, Liliana F. Soares, Joo V. Gomes, Mrio M. Freire, and Pedro R. Incio. 2014. "Security issues in cloud environments: a survey" *Int. J. Inf. Secur.* 13, 2 (April 2014), 113-170.
- [9] M. J. Sule, M. Li, G. A. Taylor and S. Furber, "Deploying trusted cloud computing for data intensive power system applications" *Power Engineering Conference (UPEC)*, 2015 50th International Universities, Stoke on Trent, 2015, pp. 1-5. doi: 10.1109/UPEC.2015.7339864
- [10] Coppolino, L., D'Antonio, S., Romano, L. Exposing vulnerabilities in electric power grids: An experimental approach (2014) *International Journal of Ittai Anati, Shay Gueron, Simon P Johnson, Vincent R Scarlata. "Innovative Tech nology for CPU Based Attestation and Sealing". In Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, HASP, volume 13, 2013. Critical Infrastructure Protection, 7 (1), pp. 51-60.*
- [11] Ittai Anati, Shay Gueron, Simon P Johnson, Vincent R Scarlata. "Innovative Technology for CPU Based Attestation and Sealing". In *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, HASP, volume 13, 2013.*
- [12] Frank McKeen, Ilya Alexandrovich, Alex Berenzon, Carlos Rozas, Hisham Shafi, Vedvyas Shanbhogue, and Uday Savagaonkar. "Innovative Instructions and Software Model for Isolated Execution" *HASP*, 13:10, 2013.
- [13] Andrew Baumann, Marcus Peinado, and Galen Hunt. 2015. "Shielding Applications from an Untrusted Cloud with Haven". *ACM Trans. Comput. Syst.* 33, 3, Article 8 (August 2015), 26 pages. DOI=<http://dx.doi.org/10.1145/2799647>
- [14] F. Schuster, M. Costa, C. Fournet, C. Gkantsidis, M. Peinado, G. Mainar-Ruiz, and M. Russinovich. "VC3: trustworthy data analytics in the cloud using SGX". In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 38-54, 2015.
- [15] PR Pietzuch, S. Arnavtsov, B. Trach, F. Gregor, T. Knauth, A. Martin, C. Priebe, J. Lind, D. Muthukumaran, D. O'Keeffe, M. Stillwell, D. Goltzsche, D. Eyers, K. Rüdiger, C. Fetzer. "SCONE: Secure Linux Containers with Intel SGX". *USENIX*. 2016