Paper

SHaPe: A Honeypot for Electric Power Substation

Kamil Kołtyś and Robert Gajewski

Research and Academic Computer Network (NASK), Warsaw, Poland

Abstract—Supervisory Control and Data Acquisition (SCADA) systems play a crucial role in national critical infrastructures, and any failure may result in severe damages. Initially SCADA networks were separated from other networks and used proprietary communications protocols that were well known only to the device manufacturers. At that time such isolation and obscurity ensured an acceptable security level. Nowadays, modern SCADA systems usually have direct or indirect Internet connection, use open protocols and commercial-off-the-shelf hardware and software. This trend is also noticeable in the power industry. Present substation automation systems (SASs) go beyond traditional SCADA and employ many solutions derived from Information and Communications Technology (ICT). As a result electric power substations have become more vulnerable for cybersecurity attacks and they need ICT security mechanisms adaptation. This paper shows the SCADA honeypot that allows detecting unauthorized or illicit traffic in SAS which communication architecture is defined according to the IEC 61850 standard.

Keywords—cybersecurity, IEC 61850, honeypots, SCADA.

1. Introduction

An electric power substation is a part of a critical infrastructure, an electrical grid, which delivers essential services that many industry sectors and millions of individual consumers depend on. Substations are used to distribute electrical energy to consumers, transform voltage to different levels, supervise and protect the distribution network. In the modern substations theses functions are performed with the support of a substation automation system (SAS).

SAS realizes common tasks of a Supervisory Control and Data Acquisition (SCADA) system and also provides additional features enhancing operation and maintenance efficiency, e.g. alarm processing or substation integration [1]. Figure 1 presents a typical architecture of SAS that is divided on three levels: station, bay and process. In the station level there is a so-called Human Machine Interface (HMI) used to control, supervise and monitor the substation, a workplace for engineering and configuration purposes and interfaces for the remote communication, e.g. with a control center. The bay level consists of control, protection and monitoring units of each bay and the process level provides devices that directly interface the primary substation equipment, i.e. smart sensors, actuators.

The devices in the bay and process levels are mostly intelligent electronic devices (IEDs). IED is a device that implements particular function in a substation and has a microprocessor and communication ports to be able to transmit data and execute control commands. The examples of IEDs are circuit breakers, voltage regulators, protection relays and Programmable Logic Controllers (PLC). Big substations may have more than 100 bay level IEDs and a similar amount of process level IEDs. Those IEDs are usually made by different vendors. To provide interoperability between them the IEC 61850 standard defining common communication architecture has been proposed.

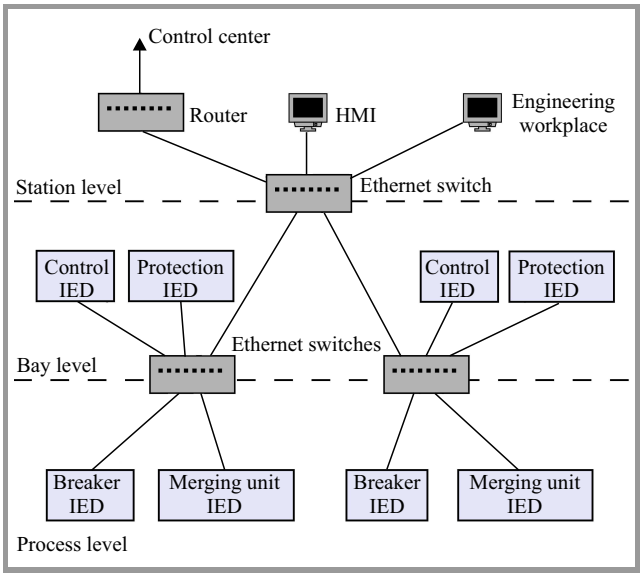

Fig. 1. A typical architecture of SAS.

In most countries the electrical grid contains typically very few substations and a failure of one of them may have severe consequences. Thus any single substation has to be carefully protected. In particular, a special attention should be devoted to the cybersecurity of SAS. The application of Ethernet and other Information and Communications Technology (ICT) solutions, as indicated by IEC 61850, makes SAS more exposed to cyberattacks.

A disclosure of cybersecurity incidents in SCADA systems confirms that they are not free of security issues. The prominent example of an attack on the SCADA system was an attempt to sabotage Iran's nuclear project by means of a computer worm known as Stuxnet. Stuxnet was released in 2009. Chen *et al.* in [2] show an overview of the Stuxnet's architecture. They point on the considerable effort needed to develop such a malware as well as on the fact that the attack would not be possible to succeed without insider knowledge and the support from a large team of

experts. The Stuxnet targeted attacks are able to penetrate into the isolated part of the SCADA system that were traditionally separated from the parts connected to the Internet. Stuxnet contains modules that attack PLCs in the target system and may cause physical damage to the equipment.
Fortunately, the awareness of cybersecurity threats is growing. According to the recent ICS-CERT report [3] in 2014 there were reported 232 incidents and 167 vulnerabilities concerning SCADA systems. It is widely recognized that the protection based on network isolation and an obscurity of proprietary communication protocols is no longer suitable for today's such systems. From the analysis of SCADA security standards presented in [4] it results that the most important cyber countermeasures are authentication and cryptography. On the other hand the most frequently mentioned threat in those standards is a malicious code. The leading SAS vendors try to address these security issues offering their products with additional security features adapted from ICT systems such as firewalls, antivirus software, advanced account management systems or intrusion detection and prevention systems supporting SCADA signatures.
A useful security mechanism that is becoming more popular in ICT systems are honeypots. As defined by Spitzner [5] a honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource. The honeypot is a trap for the attackers. One of the honeypot's aim is to maintain the attacker's interest and thus observe the attack methods. This way previously unknown attack methods can be revealed and analyzed to improve the system security. Honeypots surely can help to better protect the SCADA systems. Their application is considered in the one of big research project concerning the cybersecurity of critical infrastructures [6].
This paper presents the honeypot named SHaPe that is able to emulate any IED conforming to the IEC 61850 standard. SHaPe can be used to detect unauthorized or illicit traffic in SAS, which communication architecture is defined according to IEC 61850.
The remainder of the paper is organized as follows. In Section 2 the related work concerning SCADA honeypots is discussed. In Section 3 the main principles of the IEC 61850 standard are presented. Section 4 describes the SHaPe honeypot. Finally, Section 5 summarizes the paper.

2. SCADA Honeypots

The literature mentions only few honeypots designed especially for SCADA systems. Each of them belongs to one of the two traditional honeypot classes [7]: low-interaction or high-interaction. A high-interaction honeypot usually uses a real resource and let an attacker to interact with it, e.g. log into the operating system. A low-interaction honeypot operates by emulating a resource or some part of it making an attacker convinced that he interacts with the real resource. On the one hand the high-interaction honeypot is able to induce and thus detect any type of attack against the particular resource while the efficiency of the low-interaction honeypot is limited by the accuracy of the emulation. On the other hand the low-interaction honeypot is usually easier to deploy and maintain and involves a lower risk of the honeypot to become compromised.

One of the first initiatives concerning SCADA honeypots is the SCADA Honeynet Project [8] that was started in 2004. It aims to create a SCADA honeypot based on the low-interaction honeypot Honeyd [9]. Honeyd simulates a number of network protocols such as HTTP, SMTP and FTP but it can be extended to simulate other network protocols using simple scripts. The developers of the SCADA Honeynet Project create a number of scripts emulating a PLC device with HTTP, FTP, Telnet and Modbus services. They also implement a Java applet that shows the status of a SCADA device. The project being at the proof of concept stage has not been developed since 2005.

Based on the SCADA Honeynet Project, Digital Bond [10] develops a low-interaction SCADA honeypot that emulates a popular PLC device with SNMP and all services provided by the SCADA Honeynet Project honeypot. Moreover, Digital Bond proposes a security mechanism called SCADA Honeywall. It uses IDS with special SCADA signatures to detect known attacks and is able to stop the outbound traffic from the compromised honeypots. The SCADA Honeywall can be placed in front of either a low-interaction honeypot like the one provided by Digital Bond or a high-interaction honeypot using e.g. a real PLC.

Two different honeypot systems that have been used to collect statistical data about the SCADA cyberattacks are described in [11]. One system is a high-interaction honeypot that utilizes an actual PLC device and a physical server. The PLC mimics a temperature controller in a factory and has temperature, fan speed and light settings that can be modified. The physical server that is connected with the PLC operates as a HMI and hypothetically modifies the PLC settings. The second system is a low-interaction honeypot realized on the Amazon EC2 cloud Web service. One Amazon EC2 instance is configured as a Web page emulating the interface of a water pressure station. The another Amazon EC2 instance connected with the first one simulates PLC with DNP3 and Modbus services.

Another low-interaction SCADA honeypot emulating PLC is presented in [6]. It implements three communication protocols: Modbus, FTP and SNMP. Moreover it has a special module for detecting probing activity at the remaining TCP ports. The honeypot also provides additional features such as filtering and aggregating the security events.

One of the latest SCADA honeypots is Conpot [12] on which work began in 2013. Conpot is a low-interaction honeypot that at the default configuration emulates Siemens SIMATIC S7-200 PLC. It provides an implementation of Modbus and SNMP. The response times of emulated services can be artificially delayed to mimic the behavior of a system under constant load. Conpot can be deployed with a custom HMI. It is an open source software that can

be easily extended to emulate more complex SCADA systems. The project is actively developed under the auspice of the Honeynet Project.
At the end, it should be noted that beside the aforementioned typical SCADA honeypots there are other more general honeypot solutions that may be employed to protect SCADA systems. For example, GhostUSB [13] is a low-interaction honeypot that emulates a USB storage device. Although it does not focus on the SCADA network protocols it can be used in the SCADA system to detect malware that propagates through USB devices, e.g. Stuxnet.
Concluding, the SCADA honeypots known in the literature allow monitoring traffic involved with a HMI and typical PLC devices. They focus on the traditional SCADA communication protocols such as Modbus, SNMP, FTP and HTTP. Taking into account that these protocols are not included in the IEC 61850 standard none of the existing SCADA honeypots is suitable for modern SASs compliant with this standard.

3. Overview of IEC 61850

IEC 61850 is an international standard that defines layered communication architecture for a SAS to provide an interoperability between IEDs from different vendors. The communication architecture is based on abstract information and service models.

3.1. Information Model

The IEC 61850 information model is presented in [14]. It is an object oriented model that specifies a set of basic data types and data objects with strict naming conventions. The data objects have a fixed hierarchical organization illustrated in Fig. 2.

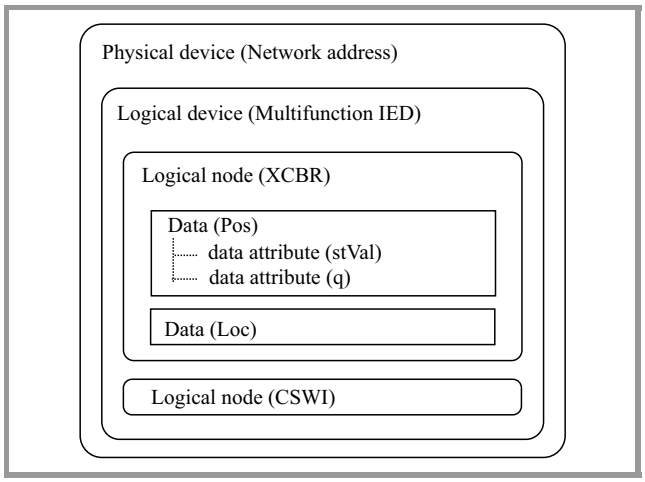

Fig. 2. The IEC 61850 information model.

At the top level of the hierarchy there is a physical device that represents an IED connected to a SAS network. The physical device is identified by its network address. It may contain one or more logical devices. The logical devices are used to form a group of some power system functions which are defined as logical nodes. Typically, the physical device has one logical device. However the possibility of having multiple logical devices allows a single physical device to act as a proxy or gateways for several IEDs.
The logical nodes contained in the logical device are the key objects in IEC 61850 representing the smallest entities of a SAS functionality used to exchange information between IEDs. A logical node is a named grouping of data objects that are logically related to the specific function. IEC 61850 defines more than 100 kinds of logical nodes covering the most common applications of SAS equipment. They are classified into 19 groups. The names of all logical nodes from the same group begin with the same character. For example the logical node XCBR that is used to model switches with short circuit breaking capability belongs to the Switchgear group which all logical nodes have names beginning with the letter X.
The semantic of the logical node is defined by its data and data attributes. Each data in the logical node has a unique name determining its purpose. IEC 61850 specifies about 500 data types with different semantic definitions. A data object may have multiple data attributes each one having name and attribute type. Data attribute names are standardized and carry specific semantic. For example the logical node XCBR has several data objects, e.g. Pos that describe a position of the circuit breaker, Loc indicating a switchover between local and remote operations or OptCnt representing an operations counter. In turn, the data object Pos has many data attributes, e.g. stVal representing a position of the real breaker that is an enumerated type taking one of the following values: intermediate-state, off, on or bad-state.

3.2. Service Model

The IEC 61850 service model is described in [15]. Like the information model it is also object oriented. The service model defines several classes with related services. The class GenServer represents the external behavior of a device. Each GenServer object contains one or more instances of GenLogicalDeviceClass class. GenLogicalDeviceClass together with three other classes GenLogicalNodeClass, GenDataObjectClass and GenDataAttributeClass represent the generic logical device, logical node, data and data attribute, appropriately. GenDataAttributeClass has an important property named functional constraint that indicates what services can be performed on the particular data attribute. For example value ST of the functional constraint means that the data attribute represents status information whose value may be read, but cannot be written.
In the service model there are also defined functional constraint data and functional constraint data attribute. The functional constraint data is an ordered collection of data attributes of the data object having the same functional constraint. The functional constraint data attribute is a data attribute having the specific functional constraint. An ordered set of elements being either a functional con-

straint data or a functional constraint data attribute is called a data set and is represented by the DATA-SET class. Data sets allow for more efficient information exchange between IEDs.
The services related with the aforementioned classes are the following:

- GenServerClass:
  - GetServerDirectory: retrieves a list of all logical devices;
- GenLogicalDeviceClass:
  - GetLogicalDeviceDirectory: retrieves a list of all logical nodes;
- GenLogicalNodeClass:
  - GetLogicalNodeDirectory: retrieves a list of all instances of a given object class,
  - GetAllDataValues: retrieves a list of all data attribute values (optionally having a given functional constraint) of all data objects;
- GenDataObjectClass:
  - GetDataValues: retrieves a list of all data attributes values,
  - SetDataValues: sets a value of a given functional constraint data or functional constraint data attribute,
  - GetDataDirectory: retrieves a list of all data attribute names,
  - GetDataDefinition: retrieve a list of all data attribute definitions (names, types and functional constraints);
- DATA-SET:
  - GetDataSetValues: retrieves a list of the values of all data attributes of the data set,
  - SetDataSetValues: sets values of all data attributes of the data set,
  - CreateDataSet: creates a data set with a given list of members being either a functional constraint data or a functional constraint data attribute,
  - DeleteDataSet: deletes a given data set,
  - GetDataSetDirectory: retrieves a list of all data set members.

IEC 61850 defines also other classes with different services that are described in detail in [15]. All services are divided on several categories. For example one category contains services supporting the device self-description. Another category is related with fast peer-to-peer exchange of status information between IEDs and yet another involves the control of an IED.

3.3. Mapping to MMS

The objects defined in the information and service models are independent of any protocol stack. However to enable real communication between IEDs these abstract objects need to be implemented in a form that can practically operate in a SAS network. IEC 61850 has established that the Manufacturing Message Specification (MMS) protocol over TCP/IP should be used for this purpose. Nonetheless another protocol can be chosen in the future to follow the evolution in ICT.
MMS is a public ISO 9506 standard that specifies the ways in which real time process data and supervisory control information is transferred between networked devices and computers. The key element of MMS is a Virtual Manufacturing Device (VMD) that models an MMS device (server) from the viewpoint of an MMS client. The VMD defines the objects (e.g. variables, domains) that are contained in the MMS server, the services (e.g. read or write a variable) a client can use to access or manipulate the objects and the behavior of the server upon receipt of those service requests. Tables 1 and 2 present the mapping of the IEC 61850 objects and services to MMS as defined in [16].

Table 1
The mapping of IEC 61850 objects to MMS objects

IEC 61850 object	MMS object
GenServerClass	VMD
GenLogicalDeviceClass	Domain
GenLogicalNodeClass	Named variable
GenDataObjectClass	Named variable
DATA-SET	Named variable list

Table 2
The mapping of IEC 61850 services to MMS services

IEC 61850 service	MMS service
GetServerDirectory	GetNameList
GetLogicalDevice-Directory	GetNameList
GetLogicalNodeDirectory	GetNameList
GetAllDataValues	Read
GetDataValues	Read
SetDataValues	Write
GetDataDirectory	GetVariableAccess-Attributes
GetDataDefinition	GetVariableAccess-Attributes
GetDataSetValues	Read
SetDataSetValues	Write
CreateDataSet	DefineNamedVariableList
DeleteDataSet	DeleteNamedVariableList
GetDataSetDirectory	GetNamedVariableList-Attributes

According to ISO 9506 a VMD and thus every instance of *GenServerClass* must also implement the following services:

- Initiate – establishes an application association, i.e. an agreement between the MMS client and the MMS server governing their communication,
- Conclude – terminates an existing application association in a graceful manner,
- Abort – terminates an existing application association in an ungraceful manner that may result in the loss of data,
- Reject – notifies about reception of an unsupported service request,
- Cancel – cancel an outstanding MMS service request,
- Identify – obtains information and status about the MMS server.

MMS services are grouped into two categories: the services requiring confirmation (so-called confirmed services) and the services that do not require such a confirmation (so-called unconfirmed services). The confirmed services contain an invocation identifier that identifies the service instance.

3.4. Configuration Description Language

Proper substation operation requires appropriate configuration of all its IEDs. IEC 61850 provides a Substation Configuration Language (SCL) that allows to describe the substation topology, the communication system, e.g. how IEDs are connected to networks and subnetworks, how data objects are grouped into data sets. SCL also allows to describe the particular IED capabilities in terms of logical nodes and the relation between substation structure and the SAS functions represented by the logical nodes. The SCL is based on XML. Its specification is given in [17].
A configuration process of substation involves many SCL files to be created. The structure and functions of the substation is defined in a System Specification Description (SSD) file that for example may contain the required types of logical nodes and data. The configuration of all IEDs with the communication section and the substation description is included in a System Configuration Description (SCD) file. Each IED to be compatible with IEC 61850 must have an IED Capability Description (ICD) file describing its functional and engineering capabilities or an Instantiated IED Description (IID) file including its project specific configuration. Moreover, it must be able to use the SCD file to set its communication configuration that is saved as a Configured IED Description (CID) file.

4. SHaPe

The IEC 61850 standard was published in 2004 and since then it has gained popularity among electric utilities and power system authorities in many countries. It can be taken for granted that more and more substations will be upgraded to conform the IEC 61850 standard. Thus a future SCADA honeypot to be useful in the electric power industry must support the communication protocols specified in IEC 61850.

4.1. General Concept

SHaPe is a low-interaction honeypot that is able to emulate any IED compliant with IEC 61850.
The low-interaction approach allows to achieve several important goals. Firstly, SHaPe can be easily configured to emulate different devices. What is needed is to provide an ICD or IID file with the requested IED configuration. The SCL file can be prepared using some IEC 61850 configuration tool or simply obtained from the existing IED if the similar one has to be emulated by SHaPe.
Secondly, SHaPe does not require any specialized equipment or much computing resources. A typical personal computer may run several instances of SHaPe. Each instance can listen on multiple IP addresses. Taking into account that a traffic coming into honeypots is rather low one machine should be enough to deploy a farm of SHaPe honeypots emulating many IEDs of different types.
Finally, SHaPe as a low-interaction honeypot involves lower risk of being compromised by an attacker than a high-interaction solution [18].

4.2. Detection Scope

SHaPe emulates IED behaviour that is involved with the MMS communication over TCP/IP connection. The generic substation events based on GOOSE or transmission of sampled values that according to IEC 61850 are mapped to other protocol stacks are not handled by the SHaPe honeypot. Nonetheless all IEC 61850 services mapped to MMS are supported and executed by SHaPe. If a service creates, modifies or deletes some object the state of the emulated IED will be accordingly updated.
SHaPe allows for detecting many important events that may appear during the communication with the emulated IED. These events are the following:

- an establishment of a TCP connection,
- a termination of a TCP connection,
- an establishment of an MMS application association (Initiate service),
- a graceful termination of an MMS application association (Conclude service),
- a receipt of a Reject service request,
- a receipt of an Identify service request,

- a receipt of a GetNameList service request,
- a receipt of a Read service request,
- a receipt of a Write service request,
- a receipt of a GetVariableAccessAttributes service request,
- a receipt of a DefineNamedVariableList service request,
- a receipt of a DeleteNamedVariableList service request,
- a receipt of a GetNamedVariableListAttributes service request,
- a receipt of an unknown MMS request before establishing an MMS application association,
- a receipt of an unrecognized MMS request after establishing an MMS application association.

Note that all IEC 61850 services are captured by SHaPe in terms of appropriate MMS services. Moreover, SHaPe detects the establishment and termination of every TCP connection and the receipt of MMS services related to VMD except Abort and Cancel services.

4.3. Monitoring Multiple Network Addresses

One instance of SHaPe can emulate an IED of a particular type. However SHaPe is able to run many copies of the emulated IED each one having assigned different IP address. In this way SHaPe allows for easily increasing the number of monitored IP addresses and thus the attack surface involved with the particular type of IED.
For each running copy of IED SHaPe maintains a separate state, i.e. the values of all data attributes in all logical nodes. The IED state can be modified by some IEC 61850 services, e.g. *SetDataValues* or *SetDataSetValues*. All MMS clients connected with the same IED copy see the modifications made by any of them. SHaPe keeps the modified state until there is no MMS client connection over a predefined period of time. After this idle time the IED state is restored to the initial one.

4.4. Implementation

SHaPe has been implemented as a module of Dionaea [19], which is a general purpose low-interaction honeypot running on the Linux platform. Dionaea has several modules that emulate different services prevalent in typical computer networks, e.g. HTTP, FTP, SMB. None of these modules can emulate a typical SCADA device. SHaPe is the first Dionaea module that is designed for SCADA networks.
Dionaea provides two useful mechanisms for their modules: a communication mechanism handling TCP connections and a logging mechanism that registers security events. Thanks to the communication mechanism SHaPe does not have to operate directly on the TCP sockets as it can handle appropriate events related to the transport communication layer, e.g. establishing a new TCP connection, terminating a TCP connection or receiving data. The events concerning the establishment or termination of a TCP connection are automatically forwarded to the Dionaea logging mechanism. For each establishment of a TCP connection the following information is registered: the TCP connection identifier, the timestamp, the source IP address, the source port, the destination IP address and the destination port. An event of the termination of a TCP connection contains the identifier of the TCP connection and the timestamp.
The Dionaea logging mechanism saves event information in a log file and optionally sends it to a specific XMPP server. SHaPe uses this logging mechanism to register events related to MMS protocol layer in the same way as the TCP layer events are registered. For each MMS protocol layer event the following information is provided: the identifier of TCP connection within the event has occurred, the event timestamp, the type and body of the MMS request in which the event has been detected. The type of MMS request indicates one of the unconfirmed services (Initiate, Conclude or Reject) or that a requested service is confirmed. In the latter case SHaPe registers additional information – the subtype corresponding to the particular confirmed MMS service and the invocation identifier of the service instance.
To handle MMS requests SHaPe utilizes library libiec61850 [20] that for integration purposes has been slightly modified.
Both Dionaea and libiec61850 are an open source software. Also SHaPe is publicly available under GNU GPL at the SHaPe project website [21].

5. Summary

In this paper the honeypot named SHaPe is proposed. SHaPe opposed to other SCADA honeypots supports the IEC 61850 standard. Thus it can be used to protect a modern SAS conforming to this standard.
SHaPe can be easily configured to emulate any IED by providing an appropriate SCL file. One SHaPe instance may listen on multiple IP addresses maintaining many copies of the particular IED. Several SHaPe honeypots allow to create a network of different IED decoys significantly increasing the chance of detecting an unauthorized or illicit traffic in SAS.
SHaPe has been implemented as a module of Dionaea which is a general purpose low-interaction honeypot. To handle MMS requests according to IEC 61850 SHaPe uses library libiec61850. SHaPe along with Dionaea and libiec61850 is an open source software publicly available under GNU GPL.

Acknowledgements

This work was supported in part by the National Centre for Research and Development (NCBiR) under the research project “The system of secure IP communication provision

for the power system management" (no. ROB 0074 03 001). The authors would like to thank Tomasz Pałka for his contribution in developing the SHaPe software.

References

[1] W. Rebizant, J. Szafran, and A. Wiszniewski, *Digital Signal Processing in Power System Protection and Control*. Springer, 2013.

[2] T. M. Chen and S. Abu-Nimeh, "Lessons from stuxnet", *IEEE Comp.*, vol. 44, no. 4, pp. 91–93, 2011.

[3] "ICS-CERT Year in Review 2014", Industrial Control Systems Cyber Emergency Response Team, 2014 [Online]. Available: https://ics-cert.us-cert.gov/Year-Review-2014

[4] T. Sommestad, G. N. Ericsson, and J. Nordlander, "SCADA System cyber security – A comparison of standards", in *Proc. IEEE Power Energy Soc. General Meet.*, Minneapolis, MN, USA, 2010.

[5] L. Spitzner, "Honeypots: catching the insider threat", in *Proc. 19th Ann. Comp. Secur. Appl. Conf. ACSAC 2003*, Washington, DC, USA, 2003, pp. 170–179.

[6] P. Simões, T. Cruz, J. Gomes, and E. Monteiro, "On the use of Honeypots for detecting cyber attacks on industrial control networks", in *Proc. 12th Eur. Conf. Inform. Warfare Secur. ECIW 2013*, Jyväskylä, Finland, 2013.

[7] L. Spitzner, *Honeypots: Tracking Hackers*. Boston, MA, USA: Addison-Wesley, 2002.

[8] V. Pothamsetty and M. Franz, "SCADA HoneyNet Project: Building Honeypots for Industrial Networks", 2005 [Online]. Available: http://scadahoneynet.sourceforge.net/

[9] The Honeyd website [Online]. Available: http://www.honeyd.org

[10] The SCADA Honeynet website [Online]. Available: http://http://www.digitalbond.com/tools/scada-honeynet

[11] K. Wilhoit, "Who's Really Attacking ICS Equipment?", Trend Micro Research, Cupertino, CA, USA, 2013.

[12] The Conpot website [Online]. Available: http://www.conpot.org

[13] The Ghost USB honeypot website [Online]. Available: http://code.google.com/p/ghost-usb-honeypot

[14] "Communication networks and systems for power utility automation – Part 7-1: Basic communication structure – Principles and models", IEC 61850-7-1, 2011.

[15] "Communication networks and systems for power utility automation – Part 7-2: Basic information and communication structure – Abstract communication service interface (ACSI)", IEC 61850-7-2, 2010.

[16] "Communication networks and systems for power utility automation – Part 8-1: Specific communication service mapping – Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3", IEC 61850-8-1, 2011.

[17] "Communication networks and systems for power utility automation – Part 6: Configuration description language for communication in electrical substations related to IEDs", IEC 61850-6, 2010.

[18] K. Gorzelak, T. Grudziecki, P. Jacewicz, P. Jaroszewski, Ł. Juszczyk, and P. Kijewski, "Proactive Detection of Security Incidents", Tech. Rep., ENISA, 2012.

[19] The Dionaea website [Online]. Available: http://dionaea.carnivore.it

[20] The libiec61850 website [Online]. Available: http://libiec61850.com

[21] The ShaPe project website [Online]. Available: https://www.assembla.com/spaces/scada-honeypot

Kamil Kołtyś received the M.Sc. and Ph.D. degrees in Computer Science from Warsaw University of Technology (WUT) in 2007 and 2012, respectively. He was a research assistant at WUT from 2011 to 2012. From 2012 to 2015 he worked as a lecturer in Institute of Control and Computation Engineering of WUT. Since 2012 he has been an associate professor at Research and Academic Computer Network (NASK). His research interests include honeypots, data analysis, graph theory and operations research.

E-mail: kamil.koltys@nask.pl
Research and Academic Computer Network (NASK)
Wąwozowa st 18
02-796 Warsaw, Poland

Robert Gajewski received the M.Sc. degree in Control Engineering Science from Warsaw University of Technology in 2002. He was designing and building IT tools to optimize operations in NUKAT union catalogue. Since 2012 he has been an research assistant at Research and Academic Computer Network (NASK). His present area of interests includes spam data analysis, data mining, and security mechanisms.

E-mail: robert.gajewski@nask.pl
Research and Academic Computer Network (NASK)
Wąwozowa st 18
02-796 Warsaw, Poland