Ontological representation of networks for IDS in cyber-physical systems

Vasily A. Sartakov

ksys labs
sartakov@ksyslabs.org

Abstract. Cyber-Physical Systems (CPSs) combine information and communication technologies and means controlling physical objects. Modern infrastructure objects such as electrical grids, smart-cities, etc. represent complex CPSs consisting of multiple interconnected software and hardware complexes. The software contained in them requires development, support, and in case of updates termination can be the target for malicious attacks. To prevent intrusion into networks of cyber-physical objects one can use Intrusion-Detection System (IDS) that are widely used in existing noncyber-physical networks. CPSs are characterized by formalization and determinacy and it allows to apply a specification-based approach for IDS development.
This paper is devoted to IDS development using the ontology-based representation of networks. This representation allows to implement both at the software level - by means of comparing movement of network traffic with its model, and at the physical level - by means of controlling connections of network devices. Ontological representation provides a model of network which is used for creation specifications for IDS.

Keywords: networks, ids, ontology

1 Introduction

Modern approaches to the development of critical infrastructure elements are characterized by a high degree of penetration of information and communication technologies. For example, an up-to-date object of energy infrastructure is a substation and it represents a cyber-physical object which main elements are data transmission facilities and control elements. In other words, an infrastructure object represents a network of interacting intellectual components, each element of which can be implemented as a separate hardware and software complex having its own software and implementing a specific function.

Individually, hardware and software complexes include application and system software that shall be updated with the course of time (for some software solutions) for the purposes of security improvement. In case software is not updated, CPS elements may be exposed to invasions and used by malicious users to affect other components of the network because of vulnerabilities.

This work is devoted to the development of a network instruction detection system. Among many types of attacks aimed to cause abnormal operation of critical infrastructure elements we are focused on network intrusions. Our model of an intruder is based on the assumption that intrusion is carried out inside the network by means of abnormal affection of some elements of the critical infrastructure object on the others. As a consequence, the applied protection method utilizes the formalized model of the system describing all kinds of interactions of network elements and detects deviations from this model in network traffic.

2 Intrusion Detection Systems

IDSs are a widely used tool of information security provision. Coupled with active network security tools such as firewalls, these technologies allow to protect and control interaction of network components, detect malicious traffic and intrusions. There are a lot of developers and producers of such equipment on the market, new systems are being actively developed both in industry and in the academic community.

Conceptually, all kinds of IDSs can be referred to two opposite approaches [1]. The first approach is based on the analysis of network traffic and detection of specific data sequences in it. Such traffic analysis is called *signature analysis* and, consequently, the IDS is also called signature IDS. If the attack signature is correctly described, such method of instruction detection is distinguished by a high rate of actuations and a low rate of type II errors. This method requires continuous support of signature database without which it is impossible to counter instructions of known types as well as new ones and it is a disadvantage of this method.

Another approach to development IDS is based on detection of *anomalies*. Unlike the signature-based approach where specific intrusion signatures are detected in traffic, a deviation in the behavior of the controlled system is detected when dealing with anomalies. This approach applies statistical profiling, machine learning methods, etc. to detect "normal" behavior and deviations. The complexity of development of IDS of this type is a correct description of the model of "normal" behavior. An incorrectly constructed model of behavior can result in false actuations and malfunctioning. At the same time, a correctly described system can detect both known attacks and, in some cases, new ones.

Besides detection of anomalies and attack signatures there are a few intermediate approaches that include some elements of the above mentioned ones. For example, a probabilistic approach, sometimes called statistical or Bayesian [2]. Within the framework of this approach a description of the system, its states and probabilities of transition of one state to another is generated. As for the other approach based on the specification (a specification-based approach) a model of a working network is created, i.e. it is detected what interactions are allowed in the network, then the model is expanded by means of a set of security policies, and further on, the IDS responds to deviations in the operation of the observed system from its model [3].

As a general approach for the development of IDS in critical infrastructures we use the approach based on the specification. First, as compared with other

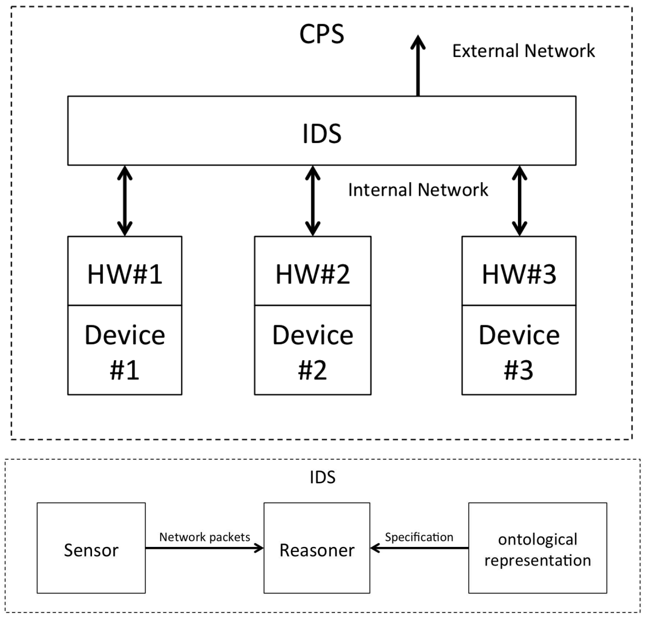

Fig. 1. IDS for CPS

methods, this method is distinguished by the lowest rate of type II errors. Moreover, unlike the signature approach, it does not require continuous support and development of signature databases and it allows to detect unknown types of attacks and intrusions [4]. Third, despite the high complexity of network specification development, the field of application allows to decrease the complexity of this work. Since we consider IDS in the context of CPSs and critical infrastructure, we assume that the structure of this infrastructure will vary slightly in course of time and thus a model of the network can be developed once and it will not require any further changes.

3 Architecture

As it was mentioned earlier, modern objects of critical infrastructure actively use information and communication technologies for components communication and remote control. Since infrastructure objects fulfill such functions as control, monitoring, impact on the physical environment, being almost fully autonomous in the process of taking decisions, we call them CPSs. Examples of such systems: power plants, smart buildings, etc.

A few peculiarities are typical for CPSs . Besides intensive communication, it is extremely important that these systems are specialized, i.e. developed specifically to be used under certain conditions and it makes the architecture of these systems to be known and determinated. In other words, all interactions of system components can be formalized and their changes are negligible with course of time. This peculiarity allows us to focus on the development of IDS based on specifications.

Another peculiarity of CPSs is working with the physical environment, and in the process of CPS development models are created, both models of the physical environment and models of the entire system. System engineering methodologies actively use digital representations of physical components in the development of industrial complexes. These models include components of different in nature, for example, people, their actions, physical devices and paper documents, but they are combined in the process of development.

These two peculiarities determine the architecture of developed IDS. An IDS is a separate component of critical infrastructure; It detects intrusions and malfunctioning of cyber-physical devices of one network.

A diagram of a network of cyber-physical devices is shown in Figure 1. As for the logistics an IDS consists of three components: a sensor, a reasoner and a network model. The sensor captures traffic. The reasoner compares network traffic with the specification obtained on the basis of the network model and performs logging of intrusions. The network model is the third component; it describes the logical and physical model of the network.

Sensor

The sensor shall access to all network traffic. Physically the sensor may be a part of communication equipment that routes traffic, or a separate network device to which traffic from switches by means of traffic mirroring is routed. The main function of the sensor is high-performance capturing of network packets and their transfer to the reasoner.

Reasoner

The reasoner receives network packets from the sensor and compares them with the network specification obtained, in its turn, from the network model. Specification is a set of rules describing known connections.

Connections are unambiguously described by two pairs *ip:port* where the first pair describes the source server of connection, and the second one - dastanition. In addition, it is important that the rule actuates just for a specific network protocol. Accordingly, the connection specification unambiguously describes the connection and this description is unambiguously based on the connections described in the network model. To describe a model of a cyber-physical network we apply ontological representation of elements of this network.

Ontological representation

As it was mentioned before, CPSs inherently represent a hybrid of a device controlling a physical process and intellectual environment in the form of hardware and software. CPSs differ from embedded ones by an available model of a physical process and interaction with the physical environment. Interaction with the physical environment and formalization of this interaction forces to use the ontological representation of this interaction, because physical and logical components of the model contact in the process of this interaction.

IDS for CPSs is a part of a critical infrastructure object. It is involved in controlling system operation and takes decisions on the separation of components of the network (in the case of operation in the mode of intrusion prevention) or informs network operators of any changes. The importance of physical representation of network components is the same as the importance of logical representation - presence of a network connection and a formal description of the connection protocol are important, but also representation of the physical location of devices is important. Physical location of devices is necessary, for example, to remove devices from the network quickly, or monitor the integrity of the physical location of the network components. As a result, we use the ontological representation of the network that allows to combine logical and physical elements of network.

Ontology

The necessity to combine logical and physical objects in the reasoner forces to use the ontological representation of these entities. Ontology is a formal, clear, precise definition of conceptualization. Conceptualization is an abstract, simplified view of the world formed for specific purposes [5]. In other words, it is possible to create a description of entities through conceptualization, and we need it to describe a network of CPS. Formalization of the same description will allow to use machine learning tools in future for semi-automatic changing of models of these networks.

To ensure the basic functionality of IDS it is necessary to describe the following entities ontologically: a server (physical, logical), a service, a network port, a protocol, a network interface, a network device, and a switch port. Moreover, it is necessary to describe relationships of these entities.

In the physical representation a server describes the location of physical communication equipment of a hardware and software complex included into the CPS. A logical server describes the entity inside which software is located. A physical server is characterized by a model of a hardware platform, its physical characteristics (for example, capacity of power supply, location), a logical server is characterized by the version of software (OS) and a set of specific services.

The service is the basic unit of communication as he fulfills network functions. The logical connection of services describes network interaction out of which may a specification for the reasoner can be eventually extracted. The service is characterized by a network port on which traffic is received and a protocol. In case the program performs client function, the logical connection of the "client" program with the "server" program shall be specified. The logical connection is required to visualize the network configuration for the network operator.

A network port, a network interface and a network device are physical and logical entities allowing to turn the logical connection of the services in the specification. A logical network interface allows to extract network address of the program, the network port - the port to which the program associated and the network device – the hardware address. Extraction of these entities allows to attach the logical service to a specific network device. The logical connection of physical network devices describes the physical connection of devices in the switch that allows to control the integrity of the physical connection of devices.

4 Evaluation

A prototype of an IDS built using the ontological representation of the network is presented in this section. To try the concept of the ontological representation we developed a sensor, a reasoner, software allowing to create a network specification, as well as network model in accordance with which specifications are created.

Sensor

A development board based on system-on-chip PPC QorIQ P2040 was used as the hardware platform. This system-on-chip allows to receive network packets directly avoiding device drivers; it helped to save on data transfer between the kernel and the user space. QorIQ system-on-chip is equipped with 4 cores; it allowed to extract network packets out of the network device in the multithreading mode and send them to the reasoner. Empirically, it was found that the maximum performance is achieved when 3 kernels are used to capture traffic (one capture thread per kernel) and one kernel - for reasoner threads. Data between the sensors and the reasoner were transferred by means of shared memory using signals to inform the reasoner of packets extraction from the network device.

Reasoner

A modified version of a multithreaded instruction detection system called Suricata [1] was used as the reasoner. Suricata is an IDS focused on the analysis of the contents of packets – searching in the contents of packets and logging. Suricata uses rules in which a pattern (a line, a data set) is described; availability of these rules is checked in all packets as well as a response to rule activation (logging, packet removal).

We do not use packet analysis as the developed IDS is built on specifications, so the basic functionality of Suricata associated with the analysis of the contents of packets was removed. At the same time we used a system of rules selection and logging used in Suricata. We used syntax to describe rules (specifications) identical to syntax of Suricata as well as the format of events representation in order not to develop separate event processing tools for the network operator.

[1] http://suricata-ids.org

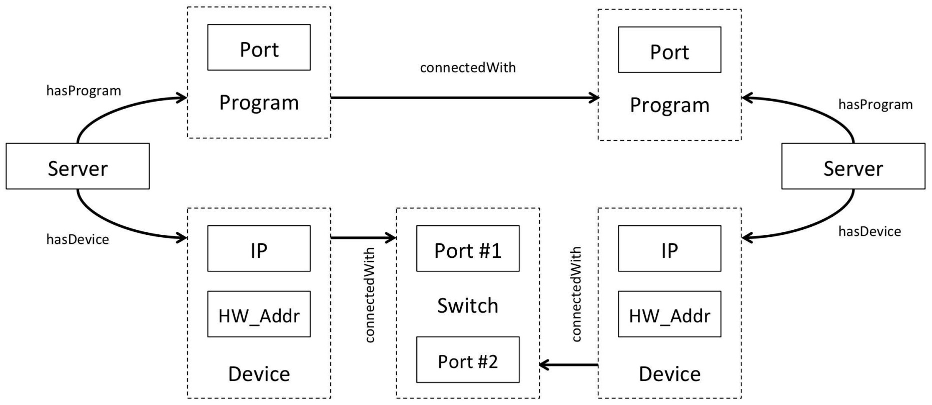

Fig. 2. Connection

Rules

Suricata uses the rules format developed for IDS Snort [6]. The rules are as follows:

```
action proto src_ip src_port -> dst_ip dst_port
                            (msg:"Message";content:"body")
```

Where *action* means action, for example, alert, *proto* is a protocol type, for example, IP, *src_ ip:src_ port* is a pair describing the source address of connection, *dst_ ip:dst_ port* is, respectively, the destination address of connection, *msg* is a message (in the case of alert) that will be logged when the rule is actuated, body is a data sequence which availability will be checked by Suricata in traffic. This syntax unambiguously describes the actuation conditions and is actively used in IDS. But as for IDS working with specifications, rules shall fulfill the opposite function - describe how it is allowed to interconnect but not vice versa.

We modified the rules processing subsystem and it allowed to describe allowed connections and not to violate the accepted rules syntax. And two rules are created for each connection two rules – on the part of the client machine and on the part of the server.

Moreover, we added binding of the machine address to IP addresses by means of inclusion of an additional keyword:

```
alert ip [3.1.1.1] any -> any any
        (msg:" Wrong hw addr";eth_src:00-00-03-01-01-01;)
```

Ontology

To describe the network model we used language of ontologies description Resource Description Framework (RDF) [7] in notation n3 [8]. RDF describes interactions of entities in format "subject-predicate-object". It allows to connect both physical and logical objects. For example, below is a description of the server identified through #S2 bound by means of predicates hasDevice, model and replica with other entities, some of which describe physical devices (for example, network device S2_eth0) and others - logical devices (for example, program s2_sps_1414).

```
@forSome <#S2> .
<#S2>          a              <Server> .
<#S2>          <name>         "S2" .
<#S2>          <hasDevice>    <#S2_eth0> .
<#S2>          <model>        <#simple1U> .
<#S2>          <hasProgram>   <#s2_sps_1414> .
```

The network description in n3 format presented above is processed by a script on Python that creates a complete set of rules based on the model. Output data of this script are a set of rules as well as a visual representation of the network and physical location of devices.

A simplified example of connection of two programs[2] located in different hardware complexes of one CPS is shown in Figure 2. The client is on the left of the diagram and the server is on the right. A network connection at the software level is presented at the top and a network connection at the hardware level is presented at the bottom. The essence of the server is connected with the essence of the program through hasProgram predicate. The program, in its turn, can be connected with another program through conectedWith entity. Both programs contain ports, but as for the client program this port is not required unlike the program serving as the server program. At the logical level the difference of the client program from the server one is detected thanks to connectedWidth predicate.

A similar situation is at the physical level. The server is connected with the essence of describing the network device through hasDevice predicate. The description of the network device contains the network address and the hardware address. Moreover, the network device is connected with the concentrator Switch port through connectedWidth predicate. The mechanism of hardware connection description is implemented through connectedWidth predicate for the purpose of its further control.

5 Related works

A lot of different IDSs have been developed in recent years. We chose several most relevant related projects out of the wide range of technologies and architectures. Since IDSs are based on specifications and ontological representation, the related projects cover the area of ontologies and IDS architectures.

The Reaction after Detection Project (RED) proposes an ontology-based approach to instantiate security policies and determine policy violations [9]. The

[2] A detailed description of the ontology, examples, network graphical representation, etc. is presented on project website http://github.com/Ksys-labs/fdnet

technology developed in the RED project provides a way to map alerts into the attack context. This context is used to identify the policies that should be applied in the network to solve the treat. The ontological representation of policies and alerts is used for this purpose. In contrast with this project we use ontological representation to describe possible interactions inside networks and we do not describe intrusions.

Another conception of ontology-based IDS was proposed by Udercoffer at. al. [10]. The idea of this project is to develop an ontology that describes intrusion in the form of anomaly. The ontology describes the attack, its consequences and the means of the attack. As another anomaly-based IDS project, the prototype demonstrates ability to detect complex attacks, but all components of the network should be installed into the IDS. In contrast with this work, our ontology is not developed to detect anomalies. We detect deviation in communications inside a network. Moreover, our prototype does not require installation of additional exemplars of IDS: we use just one IDS in each network.

The ontological representation of the networks and attacks is also provided by Frye at. al. [11]. The main goal of the paper is to present experimental project TRIDSO (Traffic-based Reasoning Intrusion Detection System using Ontology) which detects complex attacks. Complex attacks are presented by a formal method that basically provides a formal description of generic attacks and identification of new attacks. From the point of view of ontological representation our work is closely related to this work. Firstly, TRIDSO uses snort rules for traffic detection, and we use the same mechanism for the same purpose. Secondly, TRIDSO uses the ontological representation of attacks and network traffic, but we use network representation of CPS, where a network sub-system is just one component of the system. Our ontological representation does not provide ontology of an attack because we represent intrusion as deviation from the ontological model.

Ontological and semantical representations are widely used approaches for networks description. For example, Neuhaus at. al. describes [12] the ontological representation of sensors networks, while Ustalov demonstrates [13] a semantical approach to describes the configuration of a cloud environment.

6 Conclusion

This paper is dedicated to the ontological representation of the network to create specification-based IDS. Within the framework of this project we developed and created an intrusion detection system to be applied in networks of CPSs . The ontological representation of the network allowed to create formal models of CPSs that combine both physical and logical entities. IDS in current implementation require a network model based on which a network specification is created. This process requires involvement of a human. We see development of this project in automation and it means that we would like to minimize the presence of a human in the system. For this purpose means of semi-automatic creation of the ontological representation will be developed. As the ontological representation uses a machine-interpreted language, we assume that most of the work related to detection of connections and their description can be done automatically, without a human.

References

1. Zhu, B., Sastry, S.: Scada-specific intrusion detection/prevention systems: a survey and taxonomy. In: Proceedings of the 1st Workshop on Secure Control Systems (SCS). (2010)
2. Kruegel, C., Mutz, D., Robertson, W., Valeur, F.: Bayesian event classification for intrusion detection. In: Computer Security Applications Conference, 2003. Proceedings. 19th Annual, IEEE (2003) 14–23
3. Ko, C.C.W.: Execution Monitoring of Security-Critical Programs in a Distributed System: A Specification-Based Approach. PhD thesis, UNIVERSITY OF CALIFORNIA DAVIS (1996)
4. Balepin, I., Maltsev, S., Rowe, J., Levitt, K.: Using specification-based intrusion detection for automated response. In: Recent Advances in Intrusion Detection, Springer (2003) 136–154
5. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing? International journal of human-computer studies **43** (1995) 907–928
6. Roesch, M., et al.: Snort: Lightweight intrusion detection for networks. In: LISA. Volume 99. (1999) 229–238
7. Klyne, G., Carroll, J.J.: Resource description framework (rdf): Concepts and abstract syntax. (2006)
8. Berners-Lee, T., Connolly, D.: Notation3 (n3): A readable rdf syntax. W3C Team Submission (January 2008) http://www. w3. org/TeamSubmission,(3) (1998)
9. Cuppens-Boulahia, N., Cuppens, F., Autrel, F., Debar, H.: An ontology-based approach to react to network attacks. International Journal of Information and Computer Security **3** (2009) 280–305
10. Undercoffer, J., Joshi, A., Pinkston, J.: Modeling computer attacks: An ontology for intrusion detection. In: Recent Advances in Intrusion Detection, Springer (2003) 113–135
11. Frye, L., Cheng, L., Heflin, J.: An ontology-based system to identify complex network attacks. In: Communications (ICC), 2012 IEEE International Conference on, IEEE (2012) 6683–6688
12. Neuhaus, H., Compton, M.: The semantic sensor network ontology. In: AGILE workshop on challenges in geospatial data harmonisation, Hannover, Germany. (2009) 1–33
13. Ustalov, D.A.: A semantic approach for representing the cloud computing environment configuration . In: Proceedings of the 14th International Supercomputing Conference “Scientific Service on the Internet”, Moscow, MSU (2012) 706–710