

A Review of Testbeds on SCADA Systems with Malware Analysis

Uma Revisão Sobre Testbeds em Sistemas SCADA com Análise de Malware

Otávio A. M. Camargo², Julio Cesar Duarte^{1*}, Anderson F. P. Santos¹, César A. B. Andrade²

Abstract: Supervisory control and data acquisition (SCADA) systems are among the major types of Industrial Control Systems (ICS) and are responsible for monitoring and controlling essential infrastructures such as power generation, water treatment, and transportation. Very common and with high added-value, these systems have malware as one of their main threats, and due to their characteristics, it is practically impossible to test the security of a system without compromising it, requiring simulated test platforms to verify their cyber resilience. This review will discuss the most recent studies on ICS testbeds with a focus on cybersecurity and malware impact analysis.

Keywords: Malware — Industrial Control Systems — SCADA — Testbed — Industry

Resumo: Os sistemas de controle de supervisão e aquisição de dados (SCADA) estão entre os principais tipos de sistemas de controle industrial (ICS) e são responsáveis pelo monitoramento e controle de infraestruturas essenciais, como geração de energia, tratamento de água e transportes. Muito comuns e de alto valor agregado, esses sistemas têm malware como uma de suas principais ameaças e, devido às suas características, é praticamente impossível testar a segurança de um sistema sem comprometê-la, sendo necessárias plataformas de teste simuladas para verificar a sua resiliência cibernética. Esta revisão discutirá os estudos mais recentes sobre plataformas de teste de ICS com foco em segurança cibernética e análise de impactos de malware.

Palavras-Chave: Malware — Sistemas de Controle Industrial — SCADA — Plataforma de Testes — Indústria

¹ Military Institute of Engineering (IME), CEP 22290-270 Rio de Janeiro – RJ – Brazil

² Systems Development Center (CDS), QGEx - Bloco G - 2º Piso, SMU – 70.630-901 – Brasília – DF – Brazil

*Corresponding author: duarte@ime.eb.br

DOI: <http://dx.doi.org/10.22456/2175-2745.112813> • Received: 07/04/2021 • Accepted: 01/02/2022

CC BY-NC-ND 4.0 - This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

1. Introduction

Industrial Control Systems (ICS) are responsible for the control of critical strategic infrastructures, and their use has been growing lately due to the increase in automation processes, which is also followed by the growth of related cyber threats. From 2017 to 2018, the number of vulnerabilities in ICS components increased by 29%, among which more than half were rated with a CVSS¹ score equal to or greater than 7, indicating high or critical levels [1].

When dealing with security of ICS, the consequences of failures in the protection of critical infrastructures can be severe. As a result, defenses against cyber threats to ICS have become a priority in many national defense strategies. For instance, in Brazil, the Cyber Defense Doctrine, published in 2014, states that failures in strategic infrastructures constitute serious threats to national sovereignty. In consonance with this doctrine, Cyber Defense was placed as one of the three

priorities of National Defense.

In this scenario, malware is considered one of the biggest threats in the cyber environment. It is estimated that more than one million new malware samples are collected each day [2]. Process networks, which are an integral part of ICS and contain the SCADA server, are very heterogeneous and run on old operating systems that are usually not updated. This is because it is often not possible to interrupt running processes to update the system and, besides, the effects of applying updates or running antivirus applications are not known. Moreover, many specific applications cannot be ported to new operating systems.

In order to verify the resilience of ICS to cyber attacks, testbeds are used. Testbeds are platforms for testing cyber-physical systems and must capture the essence of these systems in order to be useful. In addition to being cost-effective, they are a recommended security measure because they are effective and do not harm the original systems [3].

Considering that malware can have activation mechanisms

¹ <https://www.first.org/cvss/>.

that vary according to the environment, they may not manifest entirely if the environment is not appropriate [4]. Therefore, a testbed must faithfully reproduce the original environment. Also, since malware developed for the industrial sector is generally of the zero-day² type, it is important that detection systems are able to identify threats that have not yet been reported in an automated way. Behavioral traits can be applied in malware classification, making it possible to classify even undetected threats or mutations with similar behaviors.

Malware analyses in ICS are difficult to perform because they require keeping the analyses tools hidden, besides the need of a complex platform capable of faithfully reproducing physical systems. Cumulatively, the implementation of a solution for detection in an ICS must attend to the particularities of the systems and keep up with the advances in malware development.

With the objective of analyzing the state of the art in ICS testbeds, this work reviews the most current research in the field, prioritizing those that, directly or indirectly, analyzes the effects of malware impacts.

To identify the articles that would be part of our study, supporting the analysis and review of the results, we carried out a systematic review of the literature. First, we used the following keywords in all search query providers: “malware” AND (“behavior” OR “detection” OR “survey” OR “IDS”) AND NOT (“android” OR “mobile”); “malware” AND “SCADA” AND (“ICS” OR “testbed”) AND (“machine learning” OR “deep learning”) AND NOT (“android” OR “mobile”); while making minor adjustments to accommodate some syntax differences between search providers and filtering results as of 2013. The search engines used were: Capes Journals’ Portal³; ACM Digital Library⁴; IEEE Xplore⁵; Google Scholar⁶; and DBLP⁷. Of the 560 results returned in all queries, we selected 88 articles based on their titles and abstracts, considering the existence of a consistent and relevant approach to the keywords used. We then removed articles that were published in vehicles with a low impact factor, those that used mobile or cloud platforms, and those that used testbeds unrelated to machine learning. Finally, based on these results, we add some of their references that we consider to be the most important.

This work is structured as follows: in Section 2, some of the most basic concepts are defined, such as what are ICS, SCADA systems and malware; Section 3 presents some works that approach testbeds for cybersecurity while presenting a brief discussion about the characteristics chosen for the comparison of the selected studies; in Section 4 a survey about some methods for malware detection that could be applied to zero day malware is present; and, finally, in Section 5 we present the conclusions of this work.

²those that exploit unreported vulnerabilities.

³<https://www.periodicos.capes.gov.br/>

⁴<https://dl.acm.org/>

⁵<https://ieeexplore.ieee.org/>

⁶<https://scholar.google.com/>

⁷<https://dblp.org/>

2. Basic Concepts

This section introduces the concepts of ICS, SCADA Systems and Malware.

2.1 Industrial Control Systems

ICS is a generic term that encompasses several types of control systems such as SCADA systems, Programmable Logic Controllers (PLCs), and Remote Terminal Units (RTUs). These systems are common in industrial facilities and critical infrastructure [5]. ICS have two layers of control: a physical and a cyber layer. The physical layer (i) comprises sensors, actuators, and hardware such as PLCs, which act physically in the system (opening gates, regulating voltage, pressure, etc.). The cyber layer (ii) comprises communication and information devices and their software to acquire data, develop processes and strategies and give commands to the physical layer [6]

ICS have some important characteristics, such as wide geographic distribution, constant synchronization, interaction between logical and physical infrastructures in continuous operation and system components with high useful life. Therefore, malfunctions and attacks have more tangible effects in ICS than in conventional IT systems [7].

The specificity of SCADA systems implies that most conventional IT security measures cannot be applied, and some measures can even cause harmful effects to the system instead of protecting it. Moreover, due to their always-on requirement, they cannot be updated or redesigned. Some security mechanisms such as virtual private networks (VPN) and firewalls have been successfully adopted in SCADA systems. Encryption and authentication should be used with caution not to cause unacceptable interruptions [7].

Although SCADA servers can be well protected, that does not guarantee it will also be the case with the field devices. These devices are often insecure, isolated and may have regular communication protocols, thus providing multiple access ports that are vulnerable to cyber attacks [3].

2.2 SCADA Systems

SCADA Systems are among the main tools of ICS [3]. They collect and store data from PLCs/RTUs, fully or partially manage processes and are essential for the automation and control of industrial processes. The configuration of these systems may include servers, Human-Machine Interfaces (HMI), local area networks, database servers and communication devices that allow remote control and monitoring of geographically distributed installations [8].

Figure 1 displays an example of a complete generic industrial system architecture. The upper layer contains the locations of the external access via the Internet, while the other components of the system are connected through a firewall. The corporate network group comprises workstations, printers, and application servers, among others. Attached to the corporate network is a demilitarized zone (DMZ), which consists of servers that require external access. Separated by another layer of security, there is the Control Network,

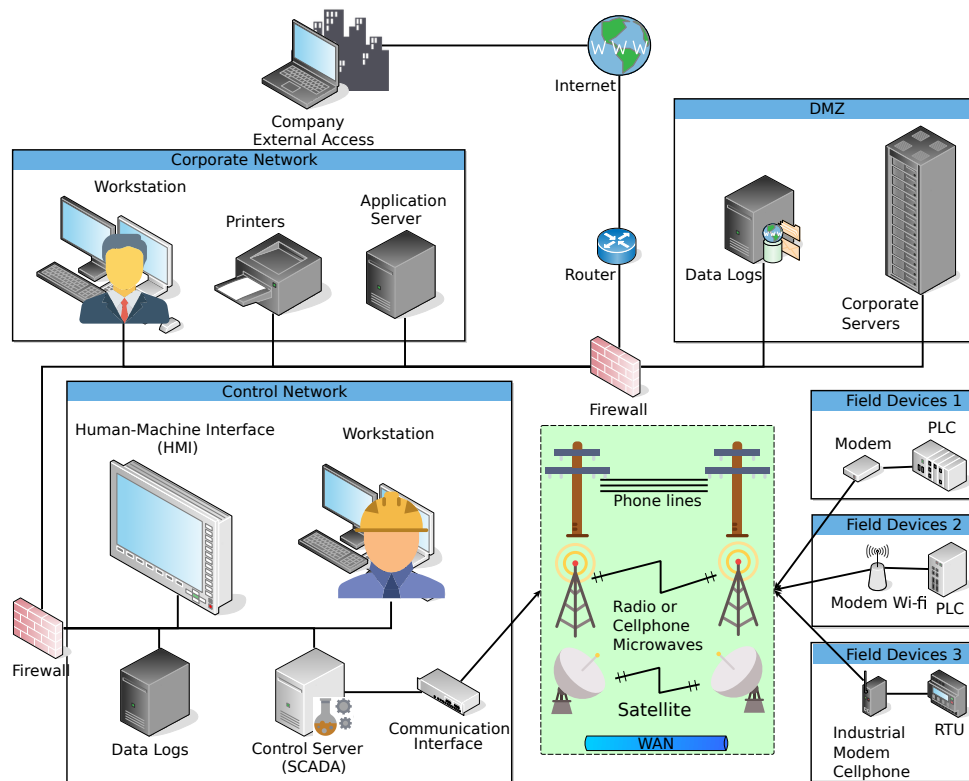


Figure 1. The architecture of an industrial system.

comprising HMI devices, workstations, and SCADA servers which are connected to the field devices, represented by Field 1 to 3, through any communication interface, such as telephone lines, microwaves, satellites, among others. Finally, PLCs and RTUs are connected directly to the industrial physical devices to monitor or change their status.

2.3 Malware

Malware (malicious software) is a generic term to describe all types of computer programs that have harmful intent [9]. These programs are used for several purposes, such as obtaining financial advantages with the use of botnets. Currently, it is possible to infect thousands of devices within a few hours after the malicious artifact is released on the network [4]. A recent example is the WannaCry ransomware, which infected more than 300,000 devices in just 24 hours [2].

The nature of criminal activity on the internet has changed and is no longer carried out by groups, but there is a diverse market for illegal digital items in support of criminal activities [10]. According to [4], the main types of malware can be divided into the following categories. It should be noted that these classes are not mutually exclusive as several malware may have behaviors and characteristics that fit into different categories.

- *Worm*. It is a program that can run independently and spread itself to other devices;
- *Virus*. It is a code snippet that adds itself to other pro-

grams, but cannot run independently, it needs a host application;

- *Trojan*. It is a software that pretends to be benign and useful, yet covertly performs malicious actions;
- *Spyware*. It is a software designed to steal information from the system of a victim that sends it to the attacker. This information can be bank accounts credentials, keystrokes, accessed pages, e-mails, etc.;
- *Bot*. It is a type of malware that allows a hacker to remotely control the device of the victim; and
- *Rootkit*. It is malware that has the ability to hide information from the system. The hiding technique itself is not malicious, however, the fact that many malware use it to hide in the system justifies such a classification.

Just like the analysis methods, malware evasion techniques are also becoming more sophisticated. Such techniques range from self-modification, dynamic code generation, metamorphosis, logic bombs, obfuscation, cryptography and packaging, to approaches that detect the presence of an analysis environment, allowing the software to only operate in specific circumstances.

It is important to analyze the behavior and techniques employed by malware, as it will directly influence the testbed implementation strategies. Also, the environments should be created as close to reality as possible, otherwise the malware may not manifest itself in its full form.

3. Testbeds

Simulations are important because it is impractical to conduct security experiments on real systems. This is due to scale, cost, and possible risks to system stability. Some vulnerabilities such as **Denial-of-Service (DoS)** attacks can cause disruptions and delays or even the entire system to crash. Implementing a real **SCADA** system just for testing, even on a small scale, is very expensive and may not be worth it due to the diversity of equipment used and its large scale. Some tested measures may be effective and the system may not be secure against certain attacks; what was designed to be a test may become a real attack [11]. Thus, to test vulnerabilities and the resilience of **ICS**, researchers turn to hardware and software simulation.

When reviewing the research literature, we found a large set of studies that presented test platforms related to **ICS** security. Nevertheless, we found very few studies discussing the detection and analysis of malware together with security in **ICS**. Most testbeds that analyze malware focus on the resilience of the physical system while most malware analysis systems do not cover **ICS** performance. In order to verify testbeds and their ability to analyze malware, we selected the most relevant attributes, which are presented in Table 1.

Besides testbeds, we also evaluated studies that verified the impact of malware on **ICS** using testbeds. We assessed the following aspects: whether real malware were used, whether malware were inserted into the corporate network and whether the impact on the physical layer was determined. We present here the most relevant details of each study.

Davis et al.[12] used the proprietary tool Power World⁸ to simulate the physical layer of an electrical power grid. For the cybernetic layer, the RINSE network simulator [13] was used, which allows large-scale real-time simulations. This approach does not provide a real IT network, nor does have a **SCADA** master server. It is possible to simulate some network attacks but, although the simulation of the physical system is quite sufficient, is not possible to truly assess the cybersecurity of the system. To measure the impact of the attacks, the packet loss rate was used as a metric.

McDonald et al.[14] presented a report from the SANDIA Laboratory⁹, in which the VCSE, a safety simulator for energy systems, is described. The network was simulated with RINSE[13] and CIPR/Sim, another simulator developed by the Idaho National Laboratory¹⁰, for simulating the resilience and protection of a critical infrastructure. It shows the ripple effect of failures in multiple infrastructures before a real event occurs. Although it is a extremely complete study, especially concerning the physical layer, and has theoretical support for testing real malware natively, no experiments with real malware were reported.

Fovino et al.[15] implemented tests on a complete testbed, with the corporate infrastructure included and physical devices connected. One of the main limitations of this study is

the lack of details about the testbed implementation, such as which operating systems were used, whether the cyber environment was virtualized or real, and whether the network was real or simulated. It also did not analyze real malware - it only simulated the behavior of five malware families using MAL-Sim [16]. Moreover, this study does not produce background traffic, although the proposed network appears to be complete, with anti-virus and intrusion detection system. MALSim is a malware behavior simulator built on top of JADE (Java Agent Development Framework). It can simulate various malware behaviors, such as mobility, duplication, and infection. All code execution can be controlled to avoid damage to the test system [16].

Chabukswar et al.[17] proposed a platform with OPNET++, Matlab/Simulink and a proprietary command and control system called C2WindTunnel¹¹, which consists of a generic modeling environment for simulations that are of the IEEE high-level architecture standard (HLA). In this study, it was not possible to analyze attacks with real malware, and a simulated Distributed Denial-of-Service (DDoS) attack was performed. As cited by Chertov, Fahmy e Shroff[18], the results of network simulators in response to DoS attacks are not reliable. This study mainly highlighted analyses of the simulated physical layer. It is a difficult solution to replicate due to the high cost of the software used.

The framework proposed by Chunlei, Lan e Yiqi[19] does not include details about how the physical part is configured, or about the capacity of the system for delays. The corporate network is simulated with the NS-2¹² and it is very detailed; however, the simulations do not allow interactions with real malware. The simulation system has an OPC server¹³, a HMI for OPC, a simulated business network, a protocol test application for **SCADA**, and real RTUs. Citect¹⁴ was used as the **SCADA** server, and real RTUs with multi-protocol capability. No attack with real or simulated malware was performed on the system, however they did perform a step-by-step attack with the elevation of privileges due to the system not having integrated malware analysis tools. This is a high cost testbed because it uses proprietary tools and real hardware.

Mirkovic et al.[20] presented the DETER project, founded by the United States Department of Homeland Security (DHS) and the U.S. National Science Foundation (NSF), a national resource for experimentation in cyber defense. The project uses *Emulab* - a public testbed of the U.S.A. in which users can access virtual machines and are able to configure their desired topology, using the applications and operating systems they choose. The project consists of a testbed for cyber simulations, which does not include tests in **ICS** yet but can be used for that.

Morris, Vaughn e Dandass[21] presented the testbed from the Mississippi State University **SCADA** laboratory. It con-

⁸ <https://www.powerworld.com/>.

⁹ <http://www.sandia.gov/>.

¹⁰ <https://www.inl.gov/>.

¹¹ <https://www.nist.gov/document/vanderbilt-c2wt-sztipanovitspdf>.

¹² <https://www.isi.edu/nsnam/ns/>.

¹³ <https://www.opcconnect.com/freesrv.php>.

¹⁴ <https://www.schneider-electric.com/en/brands/citect/>.

Table 1. Compared features

Feature	Description
Multiprotocol	Ability to operate on multiple protocols
Cyber Layer Fidelity	Have a cyber or physical layer that enables the execution of real malware
Simulated background traffic	Integration of tools to generate background traffic, such as mouse movement, file opening, network traffic, etc, necessary to prevent the testing environment from being detected and to allow real life like situations
Real corporate network	Integration of a corporate network in addition to the industrial network which can be an important vector of infection and it is crucial to test the resilience of the SCADA system to infections arising from it, whether real or virtual
Physical Layer Fidelity	Possibility to work with devices such as real PLCs/RTUs not simulated in the physical layer since attacks like DoS and buffer overflow cannot be evaluated effectively on emulated or simulated hardware
SCADA master availability	Have a functioning SCADA master server.
Real malware analysis	Allows testing, analysis and gathering of real malware information, which is necessary since it is very difficult or even impossible to simulate all malware behavior
Repetition	Ability to perform experiments automatically
Control	Ability to allow total control of the experiment, data storage and modification of variables
Resilience and Security	Enables easy post-attack recovery to ensure safe testing
Physical Multi-processes	Ability to implement several categories of physical processes simultaneously.
Interface	Friendly interface for controlling experiments which is important for professionals not related to the field of IT to operate the test platform
Documentation	The platform must have ample documentation to allow safe operation and, in the case of open source systems, its replication
Low cost	Low cost of hardware and software for implementation

sists of seven physical systems on a small scale, all real, detailed and operational. The HMI is the GE/Fanuc iFix¹⁵, which supports 3 different protocols. The system has a master SCADA implemented by the university, there is no corporate network, and it is not possible to test real malware.

Queiroz, Mahmood e Tari[11] implemented SCADASim, a testing framework on top of a simulated network in the OM-NET++¹⁶ environment. In order to simulate the physical part, MatLab/Simulink¹⁷ was used, which allows real devices to be connected to the simulator. This system can only simulate network-based attacks. Since it consists of a simulated system, it is not possible to analyze the complex interactions caused by malware. It stopped being developed in 2011, and it does not simulate hosts with different behaviors on the corporate network, such as e-mail servers, databases, etc.

The approach proposed by Genge et al.[6] supports a wide variety of physical processes, typical components of networked ICS, testing with real malware, security, resilience for studies and high reliability with real systems. The framework allows scaling up to accurately simulate large ICS with more than 100 PLCs, while being very complete and flexible. The applicability was demonstrated with two case studies - one of them addresses the influence of malware on a power plant

and the other analyzes the network conditions on a chemical plant after a cyber attack. However, no real malware was tested on this platform, just a script simulating the behavior of Stuxnet. One of the limitations of this study is the difficulty in replicability due to the use of the Emulab platform and lack of description of how the platform was configured. The focus of this study was the testing platform and not the case studies.

Ciancamerla, Minichino e Palmieri[22] tested a malware infection and its spread on the corporate network with The event simulator Netlogo¹⁸. DoS and Man-in-the-middle attacks were also simulated. The proposed testbed uses NS-2 to simulate a corporate network and Wizcon¹⁹ as the SCADA master. In order to measure the effectiveness of DoS attacks, the travel time of SCADA system packages was used as a metric. For the final impact on the smart grid, the FISIR (Fault Isolation and System Restoration) time was measured, which is the time to restore, isolate and reconfigure quickly and safely after a failure event.

Siaterlis, Genge e Hohenadel[23] presented the testbed EPIC (Experimentation Platform for Internet Contingencies), which is based on Emulab²⁰ to simulate the cyber part and on various software to simulate the physical part. Emulation with Emulab allows easy repetition, scalability, and control of

¹⁵ <https://www.ge.com/digital/products/ifix>.

¹⁶ <https://www.omnetpp.org/>.

¹⁷ <https://www.mathworks.com/products/simulink.html>.

¹⁸ <https://ccl.northwestern.edu/netlogo/>.

¹⁹ <http://www.getcontrolmaestro.com/wizcon-en.html>.

²⁰ <http://www.emulab.net/>.

experiments; however, it does not allow further analyses with the insertion of real malware. Although in theory it would be possible to make that happen, given that Emulab is composed of virtual machines, there are no analysis tools built into this solution.

Ficco, Choraś e Kozik[24] used a network simulated with OMNET++ and the physical layer was simulated with MatLab/Simulink. The simulation infrastructure was based on Portico²¹, which is an open-source implementation for HLA. Monitoring agents were implemented using JAVA Agent Development Framework (JADE). Only the simulation platform was presented, no real attacks were simulated or executed, and no case studies were presented.

Huda et al.[25] proposed a semi-supervised behavior-based approach to malware detection. It is a highly complex study that showed great results and uses a virtual machine and the Cuckoo Sandbox²² software to collect information, mainly Application Programming Interface (API) calls. It focused on cyber-physical systems (CPS) and used a database automatically updated with patterns extracted from the malware behavior. Real malware was tested statically and dynamically using Machine Learning (ML). Despite focusing on CPS, they did not test for malware on ICS. Background traffic, in addition to that provided by Cuckoo, was also not generated, and there was no corporate network. In 2018, a similar but more evolved model, based on the Internet of Things (IoT), was published [26].

Akhtar, Gupta e Yamaguchi[27] developed a testbed with OPNET++ as a network simulator and MatLab/Simulink as a simulator for real devices. It simulates the attack of malware as DDoS. Among the metrics used to verify the impacts are the voltage and the number of packets sent per second.

Jahromi et al.[28] examined the vulnerability of assisted communication against cyber attacks using co-simulation. This platform was developed using two integrated simulators: OPAL-RT, which co-simulates a microgrid and Riverbed Modeler, which consists of a network communications simulator with support for a wide variety of protocols. DoS attacks and injection of invalid data into the system were simulated. The study revealed that systems are vulnerable to these types of attacks.

Given the great diversity of existing simulation models, all of them different in many aspects, a quantitative comparison is not feasible. Therefore, we compared the studies in a qualitative way. The results are summarized in Table 2.

It should be noted that not all studies presented in Table 2 analyze malware attacks, and only one of them, Huda et al.[25], applies real malware. Only four of the studies performed experiments on virtual or real machines and another four executed tests with simulated malware and analyzed their impact on the physical layer.

Regarding networks, the comparisons between simulated and emulated testbeds made by Chertov, Fahmy e Shroff[18]

showed significant differences. These differences were due to the fact that simulators and emulators abstract a large number of system attributes and assume values involving package management, which shows these types of simulations are not reliable for security testing.

Approaches based on physical and cybernetic simulated components are cheaper and tend to have many resources, however they do not support the key features to enable experimentation with malware and real SCADA systems. Simulated systems like OPNET++ and NS-2 can model normal operations, but fail to capture the complexity of interactions between malware and real hardware.

4. Malware Detection Methods

In this section, we present some studies using ML techniques to perform malware detection. Works using ML to detect malware were researched since, as already discussed, a major threat to SCADA systems is zero-day malware. With this in mind, they become important systems that can detect threats that have not yet been cataloged. The amount of work in this area is extensive and with great results, however, many use behavioral traits that are selected manually.

The malware detection methods may be applied through dynamic or static analysis. Besides these analysis, signature or behavior-based techniques can also be applied in conjunction with the traditional approaches.

Souri e Hosseini[29] presented a literature research on malware detection using ML. The research scope is very broad and it is relevant to verify the diversity and effectiveness of the methods in the area. It presented works with dynamic detection that reached 88 to 99 % accuracy.

Signature-based methods create, through a specific heuristic, a fixed model for malware which allows its detection whenever some pattern is found during an analysis. Behavior-based methods are more flexible and can only be used based on dynamic analysis. They usually use traces collected during the execution of the malware which are then converted to features, such as system logs and API calls. ML has been used to perform the detection or classification of malicious artifacts through the use of these selected attributes that are inserted as input for the ML algorithms while generating flexible mathematical models that correspond to the malware patterns used as samples.

The researched works were separated into three categories, based on the kind of information or method that they use to conduct the detection process. The first category is for works using mixed methods, the second is for those that used volatile memory as information and the last one for those that used text mining methods. Some works can be included in more than one category.

4.1 Dynamic detection using mixed methods

Deep Sign [30] focuses on generating behavior-based “signatures”. First, it inserts malware into the virtual machine using the Cuckoo Sandbox, then all system change records are

²¹ <http://www.porticoproject.org/>.

²² <https://cuckoosandbox.org/>.

collected and a vector of words is generated for each analysis, using a Bag of words. The 20.000 most frequent terms are selected and converted to a binary bit-string format that is passed to a self-coding network to reduce dimensionality, from 20,000 to 30. These 30 real numbers are used then as representation of the signature of the *malware* behavior. This process proved to be successful in grouping all the numbers collected with ML algorithms, verifying that malware from the same family are mapped in the same group.

Fan et al.[33] uses API calls as the main features of the malware dynamic analysis, but only, 80 selected ones are used as the attributes for the chosen supervised ML algorithms. 773 malware and 253 goodware are used in order to evaluate the performance of Decision Trees, SVM and Naive Bayes classifiers, which uses a 10-fold cross validation approach. It reported a F-1 of 95% and 88%, respectively, to the algorithms Decision Tree and SVM.

Pirscoveanu et al.[34] worked with the scalable classification of malware. 42,000 samples were dynamically analyzed with the Cuckoo Sandbox and supervised ML techniques were used to perform the classification that obtained an accuracy of 98% with the Random Forest classifier. The main attributes collected were API calls. In their experiment, it is important to highlight that virtual machines were used with Windows 7, where applications of common use were installed, scripts were created to simulate background traffic and the InetSim tool was used to simulate the network services .

Andrade, Mello e Duarte[32] proposed, using sandbox tools and ML techniques, a fast and different approach for detecting malware that obtained a 90% accuracy. API calls, number of processes initiated and download counts were used as attributes collected by *Cuckoo Sandbox*. The data model used was the frequency in which terms appeared, during the analysis, in the form of a vector, into five classification algorithms: Naive Bayes, SVM, J48, CART and Random Forest. The best result again was obtained with the Random Forest algorithm, in a sample of more than 360,000 malware.

Mangialardo e Duarte[35] used the combination of static and dynamic analysis, in order to circumvent the obfuscation methods employed by malware which were classified using single Decision Trees and Random Forest algorithms. As for the chosen attributes, 9 were selected from the static analysis, based on specialized knowledge and, for the dynamic analysis, API call counts were used. It used more than 131,000 samples of malware and 2,600 of goodware. In addition to the problem of detection, a classification in malware families was also carried out. The performance, with the Random Forest algorithm, in the unified and dynamic analysis was 95.75% and 93.55%, respectively.

Some papers are not focused on presenting testbeds or on malware detection techniques like the study presented by Ajmal et al.[31] that is targeted in thread hunting techniques. In order to discover techniques, tactics, and procedures to forecast threats, scenarios for malware detection were developed. This study used SCADABR as HMI in conjunction with

Conpot, that simulated PLCs and the whole SCADA network with multiple protocols like DNP3 and Modbus. In addition, it used Cuckoo Sandbox as one of the tools to analyze unknown binaries and return data logs. As a complete testbed, it allows the analysis of real SCADA malware like stuxnet, trisis, triton, and hatman. In their experiments, more than 30 types of attacks could be analyzed and the results are about 60% faster than traditional thread hunting technologies.

4.2 Detection using volatile memory

Mohaisen, Alrawi e Mohaisen[41] introduces AMAL, an automated behavior-based malware analysis system which uses features such as the rate of use of the file system, memory, network, records, etc. From this behavior, quantifiable attributes are generated, and then used as input to ML classifiers that divides them into families of the same behavior. More than 115.000 samples were used and an accuracy of 98% was obtained using an unsupervised clustering algorithm.

Aghaeikheirabady et al.[36] collected 130 attributes from memory extracts which were manually selected. These attributes were then used as input into the ML classification algorithms through a 10-fold cross-validation approach with only 350 malware and 200 goodware. An accuracy of 98% was obtained using Naive Bayes and Decision Tree and Random Forest obtained 96.6% 98.1% accuracy, respectively.

Zaki e Humphrey[42] proposed a rootkit identification system based on changes in the volatile memory space reserved for the kernel. It uses full memory images before and after executing the malware. The Cuckoo Sandbox tool is also used to perform dynamic analysis.

Dai et al.[37] used volatile memory graphically represented to classify malware. The raw image from the memory is extracted during the dynamic analysis and then converted to a gray scale image which is, then, used to generate attributes using a histogram gradient from 1,984 malware of five different families. The ML algorithms used were multi-layer perceptron and k-NN. The method is fast and achieves an accuracy and F1 of 95.2% and 94.1% respectively with neural networks.

4.3 Detection using text mining

Mosli et al.[38] used the Cuckoo Sandbox and real machines with Windows 7 to perform the analysis of only 400 malware and 100 benign applications. From this analysis, three types of information are extracted directly from the logs of Cuckoo, API calls, registry activities and library imports, all in plain text format. With these extracts, a balanced frequency table is built with the inversion of the frequency of the terms. The samples are, then, classified as malware and not malware using the ML algorithms. An accuracy of 96% was obtained with the Stochastic Gradient Descending algorithm.

Al-Rimy et al.[40] combined behavior-based malware techniques with anomaly detection to investigate ransomware. The Cuckoo Sandbox was used to proceed with the dynamic analysis, in which each sample was executed for five seconds and then the API calls were collected. The attributes of the

Table 2. Related works

	Davis et al.[12] 2006	McDonald et al.[14] 2008	Fovino et al.[15] 2009	Chabukswar et al.[17] 2010	Chunlei, Lan e Yiqi[19] 2010	Mirkovic et al.[20] 2010	Morris, Vaughn e Dandass[21] 2011	Queiroz, Mahmood e Tari[11] 2011	Genge et al.[6] 2012	Ciancamerla, Minichino e Palmieri[22] 2013	Siaterlis, Genge e Hohenadel[23] 2013	Ficco, Choras e Kozik[24] 2017	Huda et al.[25] 2017	Jahromi et al.[28] 2020	Ajmal et al.[31] 2021
Experimentation platform analysis (Testbed)															
Multiprotocol	○	●	○	○	○	●	●	●	●	●	●	○	○	●	●
Cyber layer fidelity	○	●	●	○	○	●	●	○	●	○	●	○	●	○	○
Simulated background traffic	○	○	○	○	○	●	○	○	○	○	○	○	●	○	●
Real corporate network	○	○	*	○	○	●	○	○	●	○	○	○	○	○	○
Physical layer fidelity	○	○	●	○	●	○	●	●	●	○	○	○	○	○	○
SCADA master available	○	○	●	○	○	○	○	○	●	○	●	○	○	○	●
Allows real malware	○	○	○	○	○	○	○	○	●	○	*	○	●	○	●
Allows automated repetition	○	*	○	○	○	○	○	○	●	○	●	*	●	○	●
Allows data control	○	*	○	○	○	○	○	○	*	○	●	*	●	○	●
Resilience and security	●	●	*	*	*	*	*	*	*	*	*	*	●	●	●
Physical Multiprocesses	○	●	●	●	*	○	○	●	●	○	●	○	○	○	○
Interface	*	*	*	*	*	●	*	*	*	*	●	*	●	○	●
Documentation	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●
Low cost	●	●	○	*	○	●	*	●	○	*	●	*	●	○	●
Malware impact analysis															
Using real malware	○	○	○	○	○	○	○	○	○	○	○	○	●	○	●
External infection vector	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●
Analysis in the physical layer	○	●	●	○	○	○	○	●	●	○	○	○	○	○	○




A ● indicates that the work has the feature.

A ○ indicates that the work does not have the feature.

A * indicates that the feature could not be evaluated.

Table 3. Works related to malware detection using ML.

Work	Attributes	ML Algorithm	Accuracy (%)
Andrade, Mello e Duarte[32]	API	RF	90,00
Fan et al.[33]	API	DT	95,56
Pircoveanu et al.[34]	API	RF	98,00
Mangialardo e Duarte[35]	API	RF	93,55
Aghaeikheirabady et al.[36]	RAM	NB	98,90
Dai et al.[37]	RAM	NN	95,20
David e Netanyahu[30]*	Logs	SVM	98,60
Mosli et al.[38]	Logs	SGD	96,00
Mosli et al.[39]	RAM	SVM	91,00
Al-Rimy et al.[40]	API	SVM	99,00

Generic detection methods Use of attributes from the RAM memory Use of TF-IDF to process attributes 

Only the attributes and classifiers that obtained the best results are presented;

* - Only performed classification, not detection.

classifiers were API-grams, the combination of subsequent API calls, submitted to the calculation of the frequency of terms balanced with the inverse of the frequency in the documents. The samples were balanced with 90% of goodware and 10% of malware, using 1000 benign applications and 38,152 ransomware. Dynamic detection was performed using SVM and 10-fold cross-validation, reaching an accuracy of 99 %.

Mosli et al.[39] uses 3,130 malware and 1,157 benign applications, which, of the malware, 668 did not manifest themselves, probably because of anti-vm mechanisms. In their experiment, four systems are virtualized with Windows 7 SP1, and InetSim, in order to simulate network traffic. Images were taken from the RAM memory during the analysis with the Cuckoo Sandbox. From this memory images, the handles, which are abstract pointers used to identify and access system objects without knowing their exact location in memory, are extracted and then converted into tokens and vectors using TF-IDF. A 80% holdout cross validation was performed with the ML algorithms, obtaining an accuracy of 91% with KNN, SVM and Random Forest.

4.4 Comparative Analysis

Among the data representation techniques for classification, term frequency is one of the most used when data are in textual form or in n-grams. This technique is very efficient when dealing with a large number of attributes, such as textual data obtained from volatile memory. When the scope of research is reduced to works that detect malware from behavioral traits,

using ML and information retrieval techniques from volatile memory, the amount of research found is greatly reduced, however, their results are very promising.

It was also noticed a scarcity of studies in the area of malware detection in ICS. Huda et al.[26] emphasized that ICS have specific characteristics, which make it difficult to apply conventional detection methods. Among these characteristics are not being able to have its processes interrupted and a detection that uses many system resources can delay communication and compromise connected physical devices. The choice of using RAM memory is also due to the fact that it does not depend on the existence or not of a malware analysis system involved in the process, which can be applied to real systems. The memory image collection is a simple and low computational process while the ML training for the classifiers can be done on another computer.

Table 3 presents a summary of the main works researched in the area of *malware* detection using ML. It is important to note that there is no standard methodology for research in this area and the works are different from each other in several aspects, such as sample sizes and types, execution times, attribute types, analysis tools, processing time, among others. This requires comparisons to be made in a qualitative way, rather than quantitative. Table 3 presents a summary of the final results of each work, with the best classifiers and accuracies obtained. Those performance measures, however, should not be used as a quantitative comparison.

5. Conclusions

ICS are responsible for controlling strategic infrastructures and, given their importance, they are the target of frequent cyber attacks with military, financial, or political motivation. Also, open communication standards are increasingly being used in ICS, differently from the past when most ICS used proprietary technologies. Therefore, the vulnerabilities of conventional IT systems are also present in ICS, and the potential consequences of failures or attacks on critical infrastructures could be much more impactful.

The analysis of real malware in ICS can expose unknown vulnerabilities and security demands. In the event of an attack, it is important to know how impactful the effects could be on ICS. Testing with real malware is not a common practice, but only with this approach one can analyze the real impact of this type of attack.

A large number of studies uses approaches based on physical and cybernetic simulated components. They are cheaper, have many features, but do not support the key features to enable experimentation with real malware and SCADA. Simulated network systems can model normal operations, but fail to capture the complexity of interactions between real software, malware, and hardware.

The presentation of test platforms in ICS with real malware and impact analysis should be a priority in future studies in the field. Furthermore, they should allow the faithful reproduction of the cyber and physical layers, allowing the wide execution of the malware.

Acknowledgments

This work was partially supported by national funds through FINEP, Financiadora de Estudos e Projetos and FAPEB, Fundação de Apoio à Pesquisa, Desenvolvimento e Inovação do Exército Brasileiro, under project “Sistema de Sistemas de Comando e Controle” with reference nº 2904/20 under contract nº 01.20.0272.00.

Author contributions

All authors have contributed equally to the development of this work.

References

- [1] Kaspersky Lab. Threat Landscape for Industrial Automation Systems. *Ics Cert*, p. 1–37, 2018.
- [2] Dell Secureworks. *State of Cybercrime Executive Summary*. [S.l.], 2017.
- [3] NAZIR, S.; PATEL, S.; PATEL, D. Assessing and augmenting SCADA cyber security: A survey of techniques. *Computers and Security*, Elsevier Ltd, v. 70, p. 436–454, 2017. ISSN 01674048.
- [4] EGELE, M. et al. A survey on automated dynamic malware-analysis techniques and tools. *ACM Computing Surveys*, v. 44, n. 2, p. 1–42, 2012. ISSN 03600300.
- [5] STOUFFER, K. et al. *Guide to Industrial Control Systems (ICS) Security*. [S.l.], 2015. v. 800-82.
- [6] GENGE, B. et al. A cyber-physical experimentation environment for the security analysis of networked industrial control systems. *Computers and Electrical Engineering*, v. 38, n. 5, p. 1146–1161, 2012. ISSN 00457906.
- [7] CHERDANTSEVA, Y. et al. A review of cyber security risk assessment methods for SCADA systems. *Computers and Security*, The Authors, v. 56, p. 1–27, 2016. ISSN 01674048.
- [8] American National Standard. *Security for Industrial Automation and Control Systems Part 1: Terminology, Concepts, and Models*. [S.l.], 2007.
- [9] YE, Y. et al. A Survey on Malware Detection Using Data Mining Techniques. *ACM Computing Surveys*, v. 50, n. 3, p. 41:1–41:40, 2017. ISSN 0360-0300.
- [10] FRANKLIN, J. et al. An inquiry into the nature and causes of the wealth of internet miscreants. In: *ACM Symposium on Information, Computer and Communications Security*. [S.l.: s.n.], 2007. p. 375–388. ISBN 9781595937032. ISSN 15437221.
- [11] QUEIROZ, C.; MAHMOOD, A.; TARI, Z. SCADASim - A framework for building SCADA simulations. *IEEE Transactions on Smart Grid*, v. 2, n. 4, p. 589–597, 2011. ISSN 19493053.
- [12] DAVIS, C. M. et al. SCADA cyber security testbed development. In: *North American Power Symposium*. [S.l.: s.n.], 2006. p. 483–488. ISBN 142440228X.
- [13] LILJENSTAM, M. et al. RINSE: The Real-Time Immersive Network Simulation Environment for Network Security Exercises. *Principles of Advanced and Distributed Simulation*, v. 82, n. 1, p. 43–59, 2005. ISSN 00375497.
- [14] MCDONALD, M. J. et al. *Cyber effects analysis using VCSE*. [S.l.], 2008.
- [15] FOVINO, I. N. et al. An experimental investigation of malware attacks on SCADA systems. *International Journal of Critical Infrastructure Protection*, Elsevier B.V., v. 2, n. 4, p. 139–145, 2009. ISSN 18745482.
- [16] LESZCZYNA, R.; FOVINO, I. N.; MASERA, M. Simulating malware with MAISim. *Journal in Computer Virology*, v. 6, n. 1, p. 65–75, 2010. ISSN 17729890.
- [17] CHABUKSWAR, R. et al. Simulation of Network Attacks on SCADA Systems. In: *Workshop on Secure Control Systems*. [S.l.: s.n.], 2010. v. 1, p. 8.
- [18] CHERTOV, R.; FAHMY, S.; SHROFF, N. B. Fidelity of network simulation and emulation: A case study of TCP-targeted denial of service attacks. *ACM Transaction on Modeling and Computer Simulation*, v. 19, n. 1, p. 1–29, 2008. ISSN 10493301.

- [19] CHUNLEI, W.; LAN, F.; YIQI, D. A Simulation Environment for SCADA Security Analysis and Assessment. In: *International Conference on Measuring Technology and Mechatronics Automation*. [S.l.: s.n.], 2010. v. 1, p. 342–347. ISBN 978-1-4244-5001-5. ISSN 2157-1473.
- [20] MIRKOVIC, J. et al. The DETER project: Advancing the science of cyber security experimentation and test. In: *IEEE International Conference on Technologies for Homeland Security*. [S.l.: s.n.], 2010. v. 1, p. 1–7. ISBN 9781424460472.
- [21] MORRIS, T.; VAUGHN, R.; DANDASS, Y. S. A testbed for SCADA control system cybersecurity research and pedagogy. In: *Cyber Security and Information Intelligence Research Workshop*. [S.l.: s.n.], 2011. p. 1. ISBN 9781450309455.
- [22] CIANCAMERLA, E.; MINICHINO, M.; PALMIERI, S. Modeling cyber attacks on a critical infrastructure scenario. In: *International Conference on Information, Intelligence, Systems and Applications*. [S.l.: s.n.], 2013. p. 124–129. ISBN 9781479907717.
- [23] SIATERLIS, C.; GENGE, B.; HOHENADEL, M. EPIC: A testbed for scientifically rigorous cyber-physical security experimentation. *IEEE Transactions on Emerging Topics in Computing*, v. 1, n. 2, p. 319–330, 2013. ISSN 21686750.
- [24] FICCO, M.; CHORAŚ, M.; KOZIK, R. Simulation platform for cyber-security and vulnerability analysis of critical infrastructures. *Journal of Computational Science*, 2017. ISSN 18777503.
- [25] HUDA, S. et al. Defending unknown attacks on cyber-physical systems by semi-supervised approach and available unlabeled data. *Information Sciences*, Elsevier Inc., v. 379, p. 211–228, 2017. ISSN 00200255.
- [26] HUDA, S. et al. A malicious threat detection model for cloud assisted internet of things (CoT) based industrial control system (ICS) networks using deep belief network. *Journal of Parallel and Distributed Computing*, Elsevier Inc., v. 120, p. 23–31, 2018. ISSN 07437315.
- [27] AKHTAR, T.; GUPTA, B. B.; YAMAGUCHI, S. Malware propagation effects on SCADA system and smart power grid. In: *IEEE International Conference on Consumer Electronics*. [S.l.: IEEE], 2018. p. 1–6. ISBN 9781538630259.
- [28] JAHROMI, A. A. et al. Cyber-Physical Attacks Targeting Communication-Assisted Protection Schemes. *IEEE Transactions on Power Systems*, IEEE, v. 35, n. 1, p. 440–450, 2020. ISSN 15580679.
- [29] SOURI, A.; HOSSEINI, R. A state-of-the-art survey of malware detection approaches using data mining techniques. *Human-centric Computing and Information Sciences*, Springer Berlin Heidelberg, v. 8, n. 1, 2018. ISSN 21921962.
- [30] DAVID, O. E.; NETANYAHU, N. S. DeepSign: Deep learning for automatic malware signature generation and classification. In: *IEEE International Joint Conference on Neural Networks*. [S.l.: s.n.], 2015. v. 4, p. 1–8. ISBN 9781479919604.
- [31] AJMAL, A. B. et al. Last line of defense: Reliability through inducing cyber threat hunting with deception in scada networks. *IEEE Access*, v. 9, p. 126789–126800, 2021.
- [32] ANDRADE, C. A. B. D.; MELLO, C. G. D.; DUARTE, J. C. Malware automatic analysis. In: *Brazilian Congress on Computational Intelligence*. Rio de Janeiro, Brazil: IEEE Computer Society, 2013. p. 681–686. ISBN 9781479931941.
- [33] FAN, C. I. et al. Malware detection systems based on API log data mining. In: *International Computer Software and Applications Conference*. [S.l.: IEEE Computer Society], 2015. v. 3, p. 255–260. ISBN 9781467365635. ISSN 07303157.
- [34] PIRSCOVEANU, R. S. et al. Analysis of malware behavior: Type classification using machine learning. In: *International Conference on Cyber Situational Awareness, Data Analytics and Assessment*. [S.l.: IEEE], 2015. p. 1–7. ISBN 978-0-9932-3380-7.
- [35] MANGIALARDO, R. J.; DUARTE, J. C. Integrating Static and Dynamic Malware Analysis Using Machine Learning. *IEEE Latin America Transactions*, v. 13, n. 9, p. 3080–3087, 2015. ISSN 15480992.
- [36] AGHAEIKHEIRABADY, M. et al. A New Approach to Malware Detection by Comparative Analysis of Data Structures in a Memory Image. In: *International Congress on Technology, Communication and Knowledge*. Mashhad: IEEE, 2014. p. 26–27.
- [37] DAI, Y. et al. A malware classification method based on memory dump grayscale image. *Digital Investigation*, Elsevier Ltd, v. 27, p. 30–37, 2018. ISSN 17422876.
- [38] MOSLI, R. et al. Automated malware detection using artifacts in forensic memory images. In: *IEEE Symposium on Technologies for Homeland Security*. [S.l.: s.n.], 2016. ISBN 9781509007707.
- [39] MOSLI, R. et al. A Behavior-Based Approach for Malware Detection. *IFIP Advances in Information and Communication Technology*, Springer International Publishing, p. 187–201, 2017.
- [40] AL-RIMY, B. A. S. et al. Zero-day aware decision fusion-based model for crypto-ransomware early detection. *International Journal of Integrated Engineering*, v. 10, n. 6, p. 82–88, 2018. ISSN 2229838X.
- [41] MOHAISEN, A.; ALRAWI, O.; MOHAISEN, M. AMAL: High-fidelity, behavior-based automated malware analysis and classification. *Computers and Security*, Elsevier Ltd, v. 52, p. 251–266, 2015. ISSN 01674048.
- [42] ZAKI, A.; HUMPHREY, B. Unveiling the kernel: Rootkit discovery using selective automated kernel memory differencing. In: *Virus Bulletin*. [S.l.: s.n.], 2014. p. 239–256.