JOURNAL OF CYBERSECURITY

Journal of Cybersecurity, 2020, 1–20
doi: 10.1093/cybsec/tyaa020
Research Paper

Research Paper

Decomposition and sequential-AND analysis of known cyber-attacks on critical infrastructure control systems

Peter Maynard *, Kieran McLaughlin and Sakir Sezer

Centre for Secure Information Technologies, Queen's University, University Road, Belfast, GB BT7 1NN, UK

*Corresponding address. Queen's University Belfast, University Road, Belfast, GB BT7 1NN; E-mail: p.maynard@qub.ac.uk

Received 8 July 2019; revised 20 February 2020; accepted 14 October 2020

Abstract

We perform a detailed survey and analysis of the most significant attacks, which have targeted industrial control systems over the past decade, based on detailed incident reports from scientific and non-traditional resources. This work is the first that considers together a comprehensive set of real-world cyber-attacks with the purpose of deriving a set of common features focusing particularly on the process control network. Each attack is decomposed to provide a comprehensive overview followed by a discussion of the commonalities identified across attacks. To achieve this, each attack is modelled using Attack Trees with Sequential AND, and mapped to the industrial control system Cyber Kill Chain. We focus on the methods of intrusion rather than the identification of actors. This article can be read in two parts: first, an analysis of each attack, and secondly a discussion of the derived commonalities. The resulting commonalities can be used to develop improved detection strategies to detect modern adversarial techniques and tactics.

Key words: Networking; protocols; intrusions; attack trees; ICS; Scada

Introduction

In the decade since the emergence of the Stuxnet cyber-attack in 2010, targeted attacks against industrial systems and infrastructure have evolved from a largely theoretical concern, to a reality, with several documented cases. In the years immediately following Stuxnet, scientific literature speculated on the threats and vulnerabilities that might conceptually lead to the compromise of interconnected process control networks and cyber-physical systems. However, in more recent years, a number of incidents have been publicly documented, providing valuable insights into the compromise of real infrastructure. Critical infrastructure control systems have been demonstrated to be the target of state actors and potential espionage. Sophisticated adversaries with motivation, time, resources and domain expertise can compromise an industrial control system (ICS), as proven by malware such as CrashOverride [1] and TRISIS [2], which were both designed to target industrial equipment. The past decade has seen a rapid increase in complexity of malware in general, with a notable uptick in targeted ICS attacks [3]. This new information provides an opportunity to reflect on the methods and mechanisms by which those incidents were successful, to examine specifically cyber-physical aspects that were exploited, and to explore common attack features. By analysing these incidents together, we propose that a better shared understanding can be achieved, which may offer opportunities to develop improved detection and mitigation strategies against future attacks.

Consequently, we have performed a comprehensive survey and analysis of the most significant attacks known to have targeted ICSs, which resulted in the interruption of the physical process. A decomposition of each attack is presented based on scientific and non-traditional resources. Previous research, refs. [4–6], has tended towards focusing on identification of threat actors within the ICS domain, often as part of incident response (IR) reports on newly discovered cyber-attacks. It is also noted that existing literature often investigates attacks from the viewpoint of how an adversary was able to compromise the enterprise network, for example,

discussing how lateral movement was performed within a network [7, 8]. Such a viewpoint tends towards focusing on traditional information technology (IT) systems that, although interesting, are generally well understood.

In comparison, rather than attribution from an IR perspective, this article focuses on the actions of the attackers, particularly the technical steps that most directly resulted in disruption of the normal operation of the physical process. Although previous literature tends to view single incidents in isolation, a motivation for this work is to derive a set of commonalities by considering features across a set of incidents, with the aim to facilitate a better understanding of typical adversarial behaviour and trade craft. Therefore, this article aims to classify and understand the current ICS threat landscape based on published knowledge, and will present a common approach to analysing ICS threats with improved rigour, focusing on compromise of the process control network rather than traditional IT systems. To achieve this, we will use the ICS Cyber Kill Chain (ICS-KC) [9], in combination with Attack Trees with Sequential Conjunction [10, 11] to analyse several prominent attack incidents in detail.

The contributions of this article are summarized below:

- comprehensive survey of the most significant attacks that targeted ICS, with a focus on compromise of the process control network;
- decomposition and analysis of nine ICS intrusions, presented formally using the Attach Trees with Sequential AND (SAND) modelling approach, and analysed in the context of the ICS Kill Chain; and
- proposed set of commonalities derived from the decomposition. An intended benefit of this in particular is to facilitate further work in related research fields, such as ref. [12], allowing researchers to model threats without having ICS expertise.

The rest of the article comprises two main parts: first, an analysis and decomposition of the attacks, and secondly a discussion of the derived commonalities. The structure is as follows: Background and motivation section discusses related work and further motivations for the research; modelling methods section describes the methodology and modelling methods; modelling real-world ICS attacks section presents a comprehensive decomposition of nine ICS attacks; Discussion section describes the identification of commonalities and the last section concludes this work and discusses future challenges.

Background and motivation

ICSs

Traditionally, ICSs were designed in a way that prevented remote access. This was partly by design and partly due to the limitations of the technology at the time. Following the deployment of computerized systems into critical infrastructure, which began in the early 1980s, there has been a dramatic increase in the sophistication of interconnected technologies. Because of the long life-cycle of ICS hardware, many seemingly outdated components can coexist alongside modern components within industrial and critical systems. The development of infrastructure has often involved piecemeal additions, built up over decades, resulting in interconnected and interdependent systems blending old and new technologies.

Figure 1 shows an overview of a typical ICS network, where there are four principle network enclaves or zones: DMZ, business, supervisory control and data acquisition (SCADA) and process control. They are configured as recommended by CPNI [13] and NIST SP 800-82r2 [14]. These widely accepted standards advocate the use of network segmentation to support polices for different services and processes within enclaves, boundary protection using firewalls and network intrusion detection system to detect unauthorized communication between the enclaves. Because of the nature of industrial systems, it is common to have networks that span a large geographic area, by virtue of autonomous and semi-autonomous stations located in remote areas. Typically, there may be one or more central supervisory enclaves with multiple process control enclaves including redundant connections to the SCADA enclave. All the enclaves in the depicted infrastructure contain a virtual private network (VPN) server to allow remote administration of equipment by engineering contractors, vendors and so on, who often require access for maintenance. The process control enclave contains devices such as programmable logic controllers (PLCs) or remote terminal units that interact with the physical domain via sensors or actuators. This enclave contains a number of engineering stations that are used to program the devices. Fundamental architectures and devices relating to ICS networks are covered in detail by references such as refs. [15, 16].

Many systems were developed using fieldbus protocols, designed with no form of encryption or authentication. Protocols such as IEC 60870-5-104 (IEC104) and Modbus-TCP (Modbus) allow for plaintext communication, meaning that they are susceptible to a variety of attacks, such as man-in-the-middle, replay, command injection, etc. Because of the amount of legacy equipment, the cost of replacing existing hardware, or simply vendor inertia, many ICS operators are still rolling out systems using older protocols, particularly within the process control enclave. However, even when newer protocols are used, which utilize authentication mechanisms, these can often be poorly implemented and thus provide poor security. For example, a weakness in the encryption related to the Open Smart Grid Protocol[1] was identified by Jovanovic and Neves [17], resulting in the ability to recover the private keys thereby breaking both confidentiality and authenticity of the protocol. Open Smart Grid Protocol is used in over 4 million devices. Moreover, not only are the network protocols exploitable, but the devices themselves, particularly old ones, often contain vulnerabilities. This can be due to a lack of patching of devices, since it is typical for software patching to happen in parallel with physical maintenance windows, which unfortunately may have a cadence of several years.

Any machine that is connected either directly or indirectly to the internet, or has communications with a foreign device, is at some point likely to be exposed to malicious activities and malware. Two broad types of attacks can be considered regarding ICS networks: deliberate and accidental. Accidental attacks may be caused by simple human error, misconfiguration of a device or may be due to malware that was not specifically designed to target ICS devices. On the other hand, a deliberate attack can be due to malware that is designed to seek out ICS devices with the intention to cause specific damage. Taking this concept a step further, we can also consider targeted deliberate attacks, where malware or threat actors target the technology of a particular operator or system. A report from the ICS computer emergency response team (ICS-CERT) [3] shows the most common initial point of entry for attacks against ICS was via email phishing and waterhole attacks. This suggests attacks are not affecting ICS networks simply by chance, and there is an increasing trend

1 www.osgp.org/

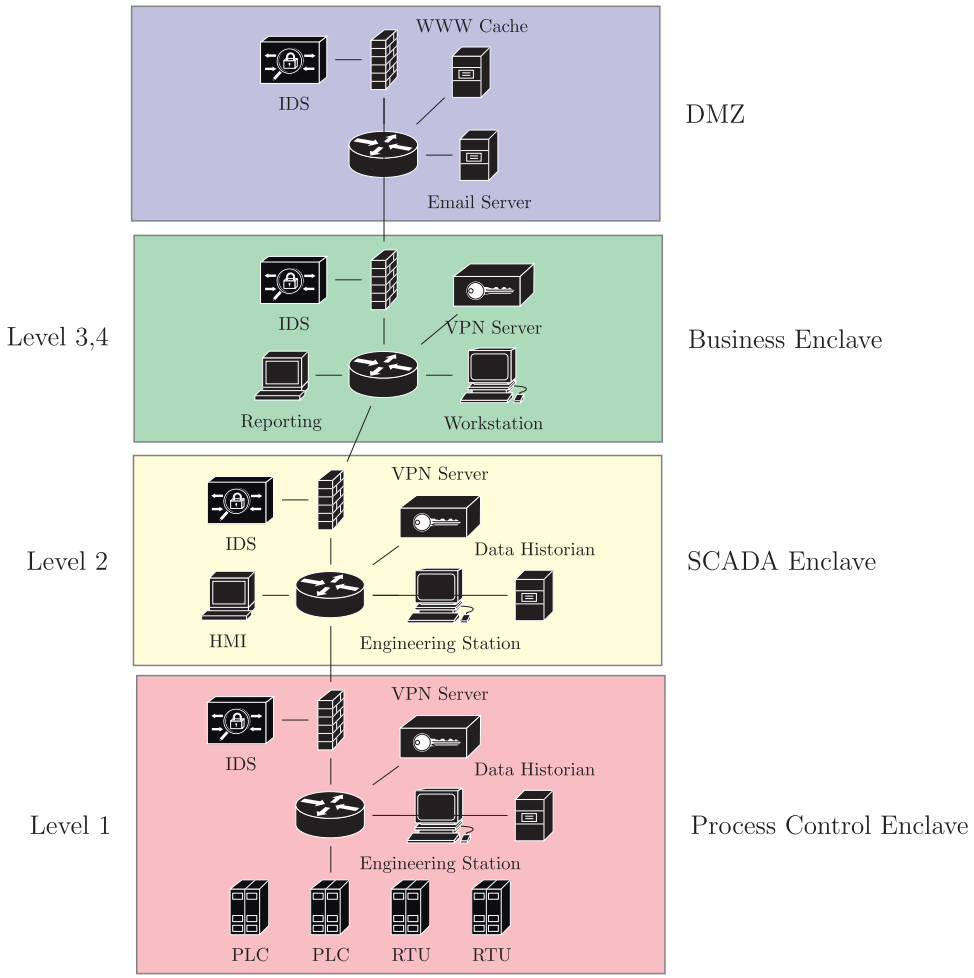

Figure 1: Traditional ICS network diagram.

towards deliberate, targeted malware campaigns, tailored for industrial systems.

Nowadays, many operators are retroactively deploying security countermeasures and monitoring equipment into control networks. Passive monitoring and agent-less systems are commonly used to detect misuse of networks. By design, passive and agent-less intrusion detection systems only store and report observed activities and will not attempt to block or disable communication of suspicious devices. However, there are also systems that can actively detect and prevent commands from executing if they are judged to be invalid. Jardine *et al.* [18], for example, use passive and active polling methods to detect intrusions based on temporal information and IP anomalies. One study by Luchs and Doerr [19] investigated the physical state of the system, as opposed to the network traffic between PLCs and control systems. In essence, such literature focuses on environmental-centric approaches, meaning they attempt to detect changes in the environment that could be a result of malicious actors.

Environmental and passive detection methods may be bypassed by a sufficiently advanced threat actor, as such in it important to understand the current techniques employed by such adversaries. Consequently, additional research is necessary which address a threat-centric approach, to better understand state of the art threats.

Threat and environmental centric approaches

The principle of a threat-centric approach is to use a corpus of knowledge derived from analysing the threat landscape, to identify indicators of compromise (IOCs), which subsequently can be used to detect threats on a network. IOCs are artefacts of data, such as an IP address for a known Command and Control (C2) server, or the binary hash of a piece of malware used to indicate a compromised host. A number of common IOCs are listed below:

- binary hashes;
- IP destination/source;
- network behaviours;
- byte sequences;
- domain names and
- protocol type.

The threat-centric approach is a targeted approach, which is important when performing IR, as it can be used to quickly identify specific intrusions to a high degree of certainty. For example, IOCs can be used to detect re-infections after a system has been recovered. However, threat-centric intrusion detection approaches alone are limited to detecting attacks that have already been analysed. Both threat and environmental centric approaches need to be considered together to secure ICS networks. MITRE has initiated a project that

combines threat and environmental-centric methods, called cyber analytics repository (CAR).[2] CAR defines a collection of abstracted behaviours based on adversarial tactics, techniques and common knowledge,[3] which represent environmental situations, but can be used in the same way as an IOC; for example, 'CAR-2013-04-002: Quick execution of a series of suspicious commands'. However, many of the abstract behaviours that have been identified require the use of host agents. Importantly, therefore, in most cases this process cannot be directly applied to ICS due to lack of agent support. In that context, it is worth noting how this article's analysis of ICS intrusions provides an in-depth review of current attacks that complement existing knowledge-bases such as MITRE abstract behaviours.

Related work

Current threat models, security specifications, and detection systems do not adequately reflect the realistic landscape of ICS threats, such as CrashOverride. Previous studies [5, 20–24] have attempted to derive common features of cyber-attacks on ICS, but operate at quite an abstracted level, resulting in a loss of the attack nuances. Bhamare *et al.* [25] critically analysed existing survey literature in ICS security. They identify a trend in ICS research to move from traditional local networks to cloud based systems, along with the use of big data analytics to detect security issues. Bhamare stated that sophisticated attacks seen in ICS have made it difficult to detect such attacks at the low level, and machine learning may be a way forward. This article aims to provide researchers attempting to use these new techniques with the data to adequately train such models. Zhu *et al.* [26] categorized the differences between ICS and traditional IT systems based on the operational priorities of each domain. They proposed that compared to traditional IT systems, ICS 0077ould prioritize: (i) timeliness; (ii) availability; (iii) integrity; (iv) confidentiality; and (v) graceful degradation. Zhu *et al.* classified attacks against ICS into attacks on hardware, software and the communication stack. Zhu *et al.* highlight the technical and domain differences between ICS and traditional IT systems, underlining that further research into domain-specific cyber-attacks on ICS is needed. Mateski *et al.* [27] considered the threat as 'a person or organization that intends to cause harm'. They stated that it is more difficult to describe a threat than it is to list it, and that it is harder still to measure them in a meaningful way. They developed a threat matrix for characterizing threats consistently and unambiguously. This approach provided a solid base for creating threat models and metrics to measure the likelihood of a specific threat. Although the approach helpfully defines a number of important methods when modelling threats, it does not allow for modelling complex threats, or decomposing specific attacks. More recently, Kotheimer *et al.* [28] investigated using the diamond model to analyse threat actors based on ICS honeypot data. However, Kotheimer noted that this resulted in gaining only a small amount of useful intelligence from their analyses, since the honeypots were unable to provide a high level of domain fidelity. These publications and related work provided insight in to commonalities across current threat actors and in some way their potential future capabilities. These findings may help to understand and mitigate specific future attacks, unlike existing standards, such as NIST SP800-53 [29] that defines a set of processes and generic controls. Looking towards real systems, the aforementioned threat commonalities can be combined with SP800-53's controls and methodology to be deployed into operations. For example, Kotheimer's analyses can be used to improve and extend IEC62351's list of threats to include contemporary real-world threats.

Attack trees were proposed [11] and allow an analyst to describe the steps required to attack a target. It closely resembles threat and fault trees that are commonly found in defence and aerospace. This similarity proved to be an advantage when working within the ICS domain since engineers are familiar with the format. Examples of attack trees within ICS include Byres *et al.* [30], who highlighted the security issues inherent to fieldbus protocols. However, due to the limited operators of 'AND' and 'OR', basic attack trees are restricted to small or relatively low complexity threats. Attack trees with sequential conjunction (SAND) [10] are an extension to attack trees, which includes an additional sequential AND in addition to the basic operators 'AND' and 'OR'. This allows for modelling more complex trees. The SAND approach is used throughout this article, and is discussed in more detail later. SAND has been applied to cyber-physical systems by Arnold *et al.* [31] to model the Stuxnet attack. Arnold applied a compositional aggregation technique to the SAND model, which translated each attack tree element into an interactive input/output Markov chain. The resulting analysis predicted the attacker would penetrate the root node with a 75% probability after 130 or more days. Time-dependent attack trees [32], are an extension to the SAND method, which allow an analyst to encode the time it might take an adversary to complete each step. Building on this, the model uses acyclic phase-type distribution, which is a class of probability distributions that can be used to approximate any other probability distribution with arbitrary precision. Acyclic phase-types are used to determine the most likely attack path based on the time it takes for an attacker to succeed. Kriaa *et al.* [17] performed a time-dependent attack tree analysis for Stuxnet, and proved that the method is suitable for modelling similar attacks. However, the approach requires complete information about the attack to be known, and thus it is only suitable for historic attacks where a full analysis has been published. Cyber security modelling language (CySeMoL) is a modelling language for system architectures coupled to a probabilistic inference engine [33, 34]. CySeMoL is designed to be used as a decision support tool, and requires full host and network information to be imported into the model. Holm *et al.* [35] performed an analysis of a substation automation architecture, that is correctly segmented with a human machine interface (HMI) running on Windows XP. The attacker is a 'hacker' attempting to connect to the HMI from the internet. The resulting analysis of this attack identified thirty different attack paths, where the top result, with a 75% success rate, was a social engineering attack to gain credentials to the remote access service.

The closet related work to this article is Grooby *et al.* [4], who modelled three advanced persistent threats (APTs) (APT1, Silent Chollima, and Molerats) that mainly targeted internet of things and ICS devices. They used the traditional Cyber Kill Chain (rather than the ICS variant) and the Diamond Model of intrusion analysis [36]. Their objective was to develop a 'cyber defence triage process' for these specific APTs. Building upon their analysis, they were able to implement a Course of Action Matrix that detailed a list of actions the defenders could take to mitigate these APTs. To help defenders better understand their potential threats, Grooby attempted to identify commonalities of each of the attacks, but did not take the next step of providing a single set of commonalties that were not specific to each attack. Finally, Grooby's focus was not directly on process

2 https://car.mitre.org

3 https://attack.mitre.org

control or SCADA networks, because of this they have missed some of the attack nuances that this article attempts to address.

Many of the modelling methods discussed aim to predict possible attack paths, the length of time an attacker might require to succeed, or the probability of success. However, they do not attempt to model the attacks to understand whether there are common features that describe how future (or previous) attacks may happen. Meanwhile, research such as ref. [28] attempts to derive common features between real-world attacks, but relies on low fidelity honeypots that lack in-depth detail relating to real intrusions into ICS networks. Consequently, this motivates the research hereby presented, which aims to address that gap by modelling real intrusions in-depth to better understand the common features or commonalities, that contribute to effective compromises in real ICS attack scenarios.

Modelling methods

The ICS-KC

The Kill Chain model was adapted from the concept of military kill chains by Lockheed Martin.[4] As the Lockheed version is not directly applicable to ICSs, a more tailored version of the model was created by SANS for ICS, called The ICS-KC, by Assante and Lee [9]. The ICS-KC is a deterministic eight-phase process, comprising two stages (Fig. 1). Stage 1, Intrusion Preparation and Execution, may be considered as espionage or an intelligence operation for traditional IT targets. In addition, in the case of ICS, an adversary would learn about the system, that is, what hardware, software, vendors, processes, etc. are used within the organization, and use this to identify specific targets. Attackers typically compromise a system and then establish a foothold within the target network. Stage 1 encapsulates these steps and does not differ from the original Lockheed Kill Chain. At Stage 2, the attackers develop targeted exploits. It is common for adversaries to have been active in the systems for many months before they execute their attack. This is a particular characteristic which differs from the original kill chain. The ICS-KC is able to capture these events within the model and provide decision support for defenders, allowing them to predict the most likely step, and subsequently deploy countermeasures at the step they believe the adversary is currently on.

Stage 1

Planning. The planning phase is where reconnaissance is performed. This is often done through observations of the target, and may uses open source information gathering tools such as Shodan and Google, along with public announcements and social media profiles of the target. Information which is of particular importance includes features of ICS equipment and technical vulnerabilities. The planning phase will aim to reveal weaknesses and identify vulnerabilities which can be used in the future phases of development and attack execution.

Preparation. Targeting and weaponization happen during this phase. Targeting is where the attackers, or their tools, identify potential victims to be exploited. Often a decision must be made to determine whether the target is worth the investment, as it may transpire that the time and effort involved is too much for the actual gain. Weaponization means preparation of a deployment method, such as a macro in an Office suite document, or compiling a malicious PDF designed for the system identified during the planning phase. It is not always necessary for both the targeting and weaponization steps to happen, for example, if VPN credentials are discovered while identifying a target, this may bypass the need for weaponizing activities.

Cyber intrusion. During this phase, the attacker attempts to gain access to the defenders' system. It comprises three steps: delivery, exploit and install/modify. The delivery step represents the method the attacker uses to interact with the defenders' system. For example, if the attacker was to use harvested VPN credentials, the VPN would be the delivery method on to the defenders' network. The exploit step represents any malicious actions the attacker takes. In the aforementioned case of the malicious PDF, the action would be to exploit a vulnerability to gain access to higher permissions. At the final step, the attacker installs malware or modifies existing capabilities for their benefit, for example, to allow remote access.

Management and enablement. After a successful intrusion, the attacker will attempt to establish some form of C2 infrastructure. This does not necessarily imply full-duplex communication between the attacker and compromised machines, and could be a simple one-way mechanism supporting low frequency messages, depending on the capabilities of the compromised systems and the requirements of the attacker.

Sustainment, entrenchment development and execution. In this phase, the attacker has many options depending on their objectives. Typically, these include lateral movement, discovering/scanning systems and data, harvesting additional credentials, installing advanced capabilities, and exfiltration of data. At this phase, the attacker might also collect data for use in future attacks.

Stage 2

Attack development and tuning. This stage starts with a phase where the attacker develops new capabilities targeted for a specific ICS implementation. In the case of the Ukrainian outages, the attackers developed custom firmware for serial to Ethernet controllers, which when flashed disabled the converters permanently. To avoid detection, development and fine-tuning of such capabilities is typically done on a test network rather than on the live compromised system. Because of the customization of the attack, there is potentially a large time lag (months or years) between Stage 1, where the attackers can have gained a foothold, to actually developing and executing a complex attack in Stage 2.

Validation. After the attackers have developed a method which they believe will work, they may validate it on the defenders' system. Alternatively, the attackers may purchase related ICS equipment with which they can assess the potential effectiveness of the attack, and to confirm it will work as expected. This requires the adversary to have domain knowledge of the industrial system they are targeting.

ICS attack. Finally, in this phase, the attacker delivers their malware, installs or modifies existing components, and executes an attack on the ICS.

4 https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

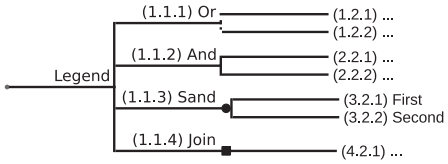

Figure 2: SAND legend.

Attack trees with sequential AND

Attack trees with sequential conjunction (SAND) were formally defined by Jhawar *et al.* [10]. SAND enhances attack trees by defining the use of an additional operator, the 'Sequential AND'. This allows child nodes to be completed in sequence, allowing modelling of more sophisticated threats compared to attack trees, while maintaining their simplistic nature and clarity. Figure 2 defines the attack tree legend, and is used throughout this article.[5] For graphical representation, note that OR is represented by a dotted line adjoining two child nodes, AND is represented by a solid adjoining line, Sequential AND is represented by a solid adjoining line with a solid circle and a joined node is represented by a solid square. Also note that child nodes are combined in a cumulative process from right to left. Numerical labels allow branches to be referred to in the main text.

Figure 3 is an example of SAND model from ref. [10], representing an analysis of a file server offering ftp, ssh and rsh services. The nodes are connected using disjunctive (OR), conjunctive (AND) and sequential conjunctive (SAND) operands. The leaves of the tree represent the actions of an attacker, with the goal of 'Become Root' at the top of the tree (presented graphically on the left). The Attack Tree shows two ways that an attack can gain root access, i.e. without authentication (1.1.1) or with authentication (1.1.2). In the first case (No-Auth), the user must gain user privileges (1.2.1) and subsequently perform a local buffer overflow attack (1.2.2). This is where the sequential AND operator is used, represented by the solid black circle, as all the steps must be completed in order for the attack to succeeded. To gain user privileges, the attacker must first exploit the FTP service (1.3.1) so that they can upload a list of trusted hosts which 'rsh' will use to allow authentication to the server (1.3.2). The second method of becoming root is to abuse a buffer overflow in both the ssh daemon (2.2.1) and the RSAREF2 library (2.2.2). These nodes are linked with AND, as each of the steps can happen in any order.

The ICS-KC and SAND approaches complement each other, since the kill chain provides a holistic outline of the adversary's steps from reconnaissance to attack execution, while the attack tree provides detailed modelling of the attack execution.

Modelling real-world ICS attacks

Based on the above methodologies, we will now perform an analysis of nine known attacks that have targeted ICS. Each attack is analysed and modelled using the SAND formalization, and then mapped to the ICS-KC to provide a holistic view of the attack. The mapping is colour coded to align with the ICS network enclaves (Fig. 1), that is, blue relates to the DMZ, green is the business enclave, yellow is SCADA, red is the process control network and grey is used for actions that are undertaken by the attacker outside the target network.

The scope of this analysis is to focus on actions specific to ICS, rather than the more traditional IT-centric threats that have been investigated elsewhere. We have adopted the terminology proposed in the PrEP framework by Herr [37], used to classify malware and cyber weapons based on the different pieces of malicious code that constitute them. Herr stated that all malwares share three fundamental components:

1. *propagation method (Pr)*: the means by which the malware propagates itself between machines;
2. *exploit (E)*: the code designed with a malicious purpose, to compromise some aspect of a software system, that allows third parties to cause unintended operations; and
3. *payload (P)*: code with a malicious purpose whose delivery and execution are the goals of any piece of malware.

Each of the components may be used in a modular fashion. Herr has used this framework in the analysis of two major examples of malware, Stuxnet and Red October. This terminology is used to avoid the vague and ambiguous definitions often used, such as worm, trojan and virus, and attempts to focus on the actual characteristics of the malware.

Figure 4 shows a timeline of each attack that will be analysed in this section, with the date based on when an analysis was first published. However, for most of the attacks, there is evidence that the victims had been compromised for a number of months or years before they were detected and analysed.

Stuxnet

Stuxnet [8] is believed to have been designed to target one specific site, which is unique in the world—Natanz uranium enrichment plant in Iran. It was designed to infiltrate the facility and cause physical damage to the system by covertly interfering with readings and the operation of centrifuge devices that are a key component of the enrichment process. It is believed that the initial entry point was via a removable storage device.

Figure 5 describes the compromise of the business enclave. Branch 1.1.1 identifies the removable storage media steps, which exploits a vulnerability within the windows autorun subsystem. This was a 0-day exploit at the time. After the initial compromise, the dropper software performed a number of privilege elevation methods (1.1.2). Once it gained appropriate privileges on the system, it executed the main module (2.2.4), that installed the root kit, and checked for any updates via P2P communication with other infected devices on the local area network, or in some cases C2.

Figure 6 shows the second part of the SAND model. This focuses on Stuxnet's activities on the SCADA network relating to propagation and payload. Note that many of the steps in the propagate branch could be detected with adequate network monitoring. The attackers were very careful to place limits in the malware to prevent it spreading widely, by using USB and local area network (2.2.1) as compared to traditional worms, which typically use indiscriminate and wide-reaching methods. Stuxnet performed complex fingerprinting, by looking for specific devices, models, and configuration details (1.4.1). It went as far as downloading logic from controllers to verify they contained the 'right' programme. As described by Langner [8] (1.4.2), it targeted two specific Siemens' PLCs, the S7-315 and S7-417. Once Stuxnet identified the controller (2.3.2), it would upload code to it and maintain a covert presence. The code

5 The python based tool 'attack-tree-generator' was used to generate the attack trees from machine readable tab indented text files. Source code and attack models can be found here: https://gitlab.com/PMaynard/attack-tree-generator

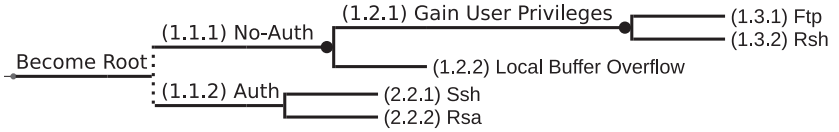

Figure 3: SAND example with the root node on the left [30].

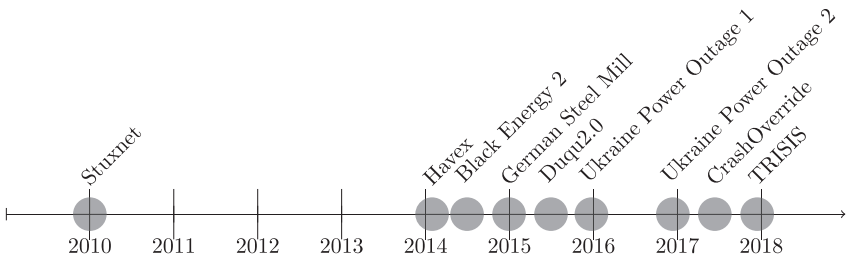

Figure 4: Timeline of ICS specific attacks which have been analysed.

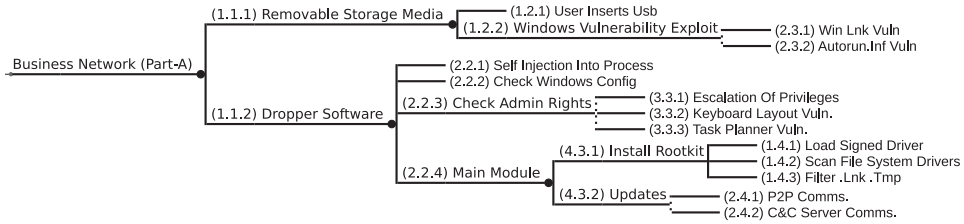

Figure 5: Stuxnet (Part A)—compromise business enclave.

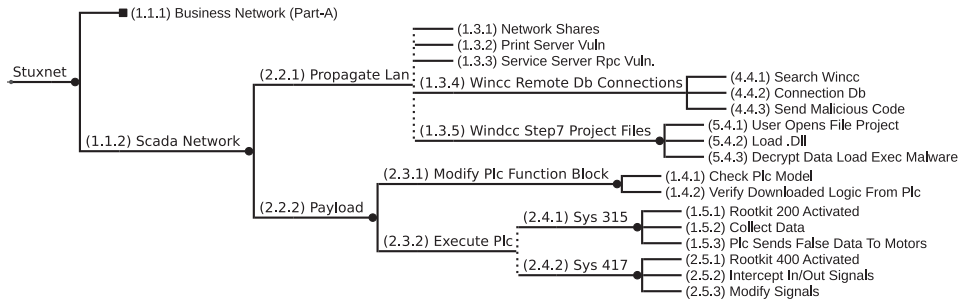

Figure 6: Stuxnet (Part B)—compromise of process control enclave.

would operate in a standalone mode, without communication to the outside world. It monitored the controller and would allow it to continue unhindered until a Stuxnet internal timer triggered. Once triggered, depending on the controller, it would:

1. *controller 315*: Halt the code, which could take 50 min to perform (Fig. 6, branches 2.5.*) and
2. *controller 417*: Perform a Man-in-the-middle attack on the I/O of controller, intercepting and replaying previously seen readings to the controller whilst providing falsified data to the I/O ports (Fig. 6, branches 1.5.*).

Figure 7 maps Stuxnet to the Stages of the ICS-KC, where it can be seen that both Stages 1 and 2 were performed. Note that a 'tick' at a Stage in the kill chain signifies the presence of that Stage in Stuxnet (as seen in later attacks, not all attacks comprise all kill chain Stages, so not all will be 'ticked'). The steps taken within Stage 1 show that the adversary had preformed detailed reconnaissance on their target and developed a targeted piece of malware that takes advantage of numerous zero-day exploits. Stage 2 shows that the attackers almost certainly developed and tested their exploit on hardware before deploying to their victim. The right of Fig. 7 shows the network levels (as introduced in Fig. 1) that each of the ICS-KC steps reach, Stuxnet propagated from the enterprise workstations down to the process control enclave, where it caused damage to the physical domain. The damage was performed by halting the process (S7-315) or by sending false information to both the process and the operators (S7-417), depending on which controller was detected.

Havex

Havex was first discovered in European electrical networks around July 2014 [38–40], and has since been attributed to the Energetic Bear/Dragonfly [41] campaign which was first identified in 2011. At the time of writing, Havex is not believed to have caused any direct damage to any specific systems, but it has been found to be enumerating networks for OLE for Process Control (OPC) devices. Havex has three propagation methods [42, 43] (Fig. 8, 1.1.1) by which it is able to bypass the business network, and penetrate straight into the SCADA enclave. In the first method, a support website for industrial routers was compromised and a legitimate patch was replaced with a trojanized version containing the Havex dropper (1.2.1). This allowed the attackers to circumvent the perimeter layers of network defences and directly access the SCADA enclave. The second method is a water hole attack (1.2.2) where specific sites were compromised to contain a malicious IFRAME, which takes advantage of Java exploits CVE-2012-1723, CVE-2013-2465 and Internet Explorer exploits CVE-2012-4792, CVE-2013-1347, to allow the dropper to be downloaded to the victim. The final method was spear-phishing (1.2.3), in which the attackers used a PDF/SWF exploit CVE-2011-0611 to infect the victim. Once a machine was compromised it sent a request to the C2 server and waited for the reply, which contained an additional plugin for the dropper.

The payload branch (1.1.2) has two steps, that is, dropper software (2.2.1) and network enumeration (2.2.2). The dropper software requests additional modules from the C2 server, and then the attack proceeds to carry out network enumeration and reporting. Both of these steps could have been prevented or detected with appropriate network monitoring. Quantitative evaluation of Havex target selection has been performed in ref. [44] in order to better detect similar malicious operations.

Figure 9 shows how Havex aligns with the ICS-KC. It completes all the Stage 1 actions, but none of Stage 2. However, Havex was able to interact with fieldbus devices, where it queried information from them. Although reconnaissance may have been the final goal of Havex, it was concerning that it was able to bypass security and communicate with devices in the process control enclave, which could have facilitated considerable harm to the ICS.

BlackEnergy

BlackEnergy is the name given to a complex and versatile backdoor malware [45, 46]. It has been used for a range of criminal activities, mainly in cases of data exfiltration where there have been apparent political motivations. The group 'Quedagh' reportedly used a BlackEnergy variant for politically oriented attacks on Ukranian

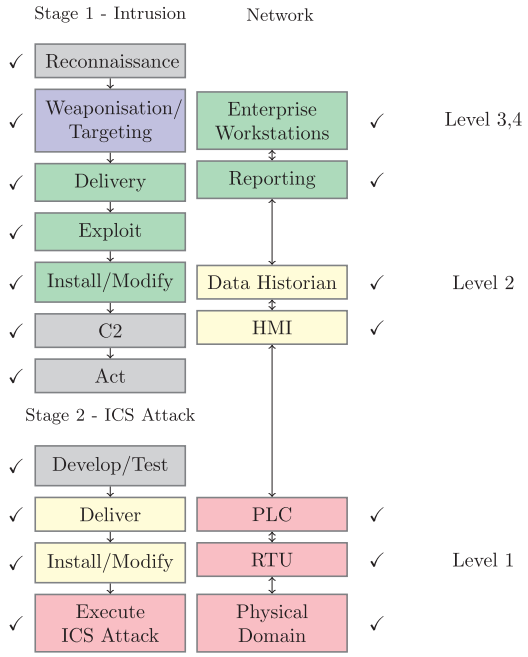

Figure 7: Mapping Stuxnet to the ICS-KC and ICS network.

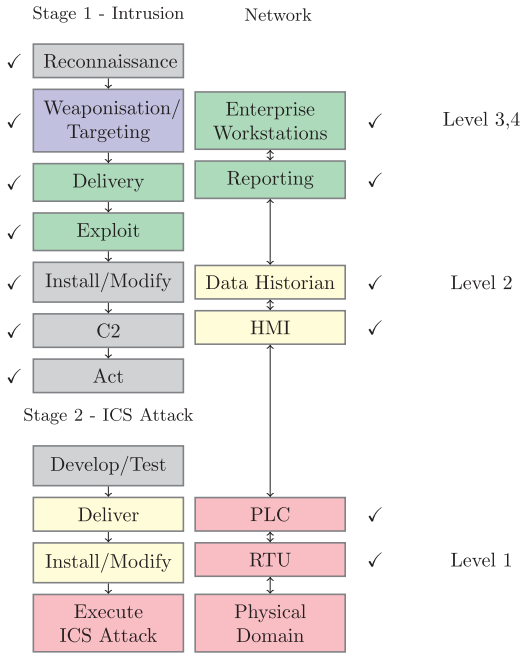

Figure 9: Mapping Havex to the ICS-KC and ICS network.

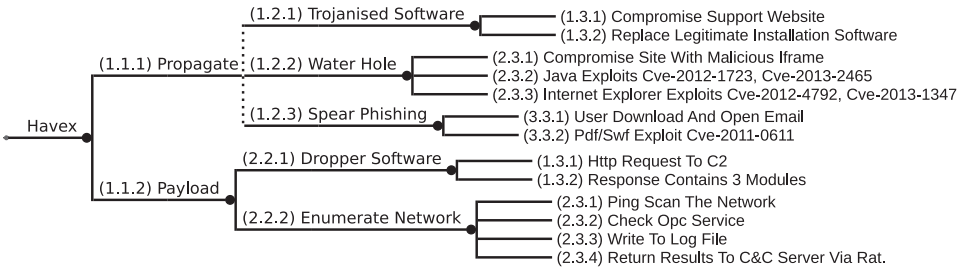

Figure 8: Havex.

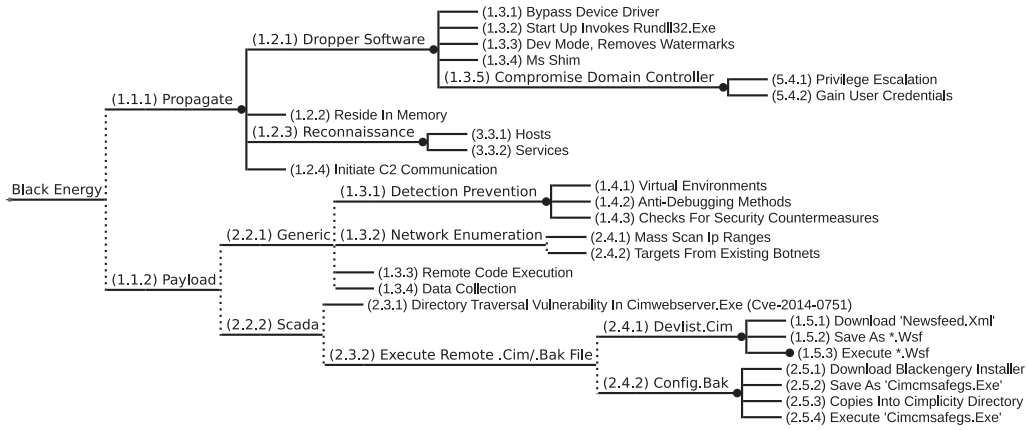

Figure 10: BlackEnergy.

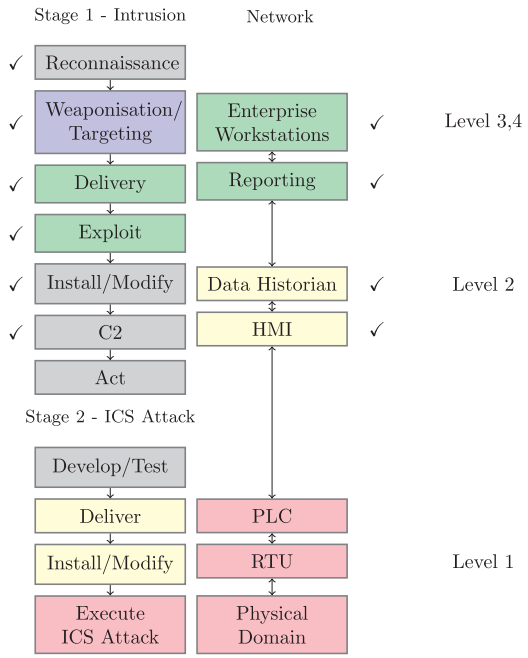

Figure 11: Mapping BlackEnergy to the ICS-KC and ICS network.

and Polish government organizations, and the Russian group 'Sandworm' reportedly used it to target organizations like the NATO alliance, energy firms and telecommunication companies. Although BlackEnergy has been widely used for information harvesting and espionage, it has also been used to target internet-facing HMIs from a range of vendors, including GE Cimplicity, Advantech/Broadwin WebAccess, and Siemens WinCC. This analysis focuses on the GE Cimplicity variation of the attack, as more information is available regarding this exploit compared to the others. However, the tradecraft required to compromise other vendors will be similar.

The two primary branches of BlackEnergy (Fig. 10) are propagate (1.1.1) and payload (1.1.2). The propagation branch identifies the steps related to the dropper software (1.2.1), reconnaissance (1.2.3) and C2 communication (1.2.4). Once established on a system, BlackEnergy scans the network for hosts and the local machine for data which can be exfiltrated (1.2.3). In addition, the dropper software has been observed to allow lateral movement and compromise of the domain controller to gain user credentials (1.3.5).

The payload branch is divided into generic (2.2.1) and SCADA (2.2.2) sub branches, which provide the malware with optional abilities that focus on generic reconnaissance and data collection, or to target specific SCADA systems such as the Cimplicity server. ICS-CERT [45] have confirmed that the attackers used automated tools to scan for specific control software, the GE Intelligent Platforms Proficy HMI/SCADA—Cimplicity. Once an instance is found (2.2.2), a known exploit is used against a directory traversal vulnerability (CVE-2014-0751) in CimWebServer.exe (2.3.1) (the WebView component) which allows remote attackers to execute arbitrary code via a crafted message to TCP port 10212, (ZDI-CAN-1623). This does not require a high level of technical skill to exploit. Once access to the system is gained (2.3.2), two files 'devlist.cim' and 'config.bak' are downloaded and executed by the HMI. The file 'devlist.cim' subsequently initiates the download and installation of BlackEnergy. Figure 11 shows that despite its sophistication, BlackEnergy reached the C2 step of Stage 1 of the ICS-KC but no further. There was no reported activity within the process control enclave, network level 1.

German Steel Mill

The German government's Bundesamt für Sicherheit in der Informationstechnik reported in their annual report of December 2014 [47] that there had been a malicious infiltration into a steel mill resulting in physical damage. Although the report did not discuss technical details, or which plant was affected, it states that the adversary used spear-phishing and appeared to be knowledgeable in the area of ICSs. The adversary infiltrated the corporate network and it is assumed they were able to pivot to the process control enclave, where they caused multiple components to fail, resulting in massive physical damage. It has been speculated that this damage could have been caused by triggering a safety instrumented system (SIS) to trip, resulting in a shutdown of a furnace. Because of the rapid change in temperature this caused significant damage to the thermal tiles inside the furnace. However, this is conjecture, and alternatively the adversary may simply have been performing reconnaissance, and triggered the event unintentionally.

Lee *et al.* [48] have written up an analysis of the attack. Despite having very limited technical information on the attack, it is a good use-case for the SAND modelling formalization as it shows the ability to model not only technical threats, but also high-level abstracted steps. Making assumptions about the system based on other attacks analysed in this article allows us to identify common features of this attack. Figure 12 shows the SAND model for the German Steel Mill

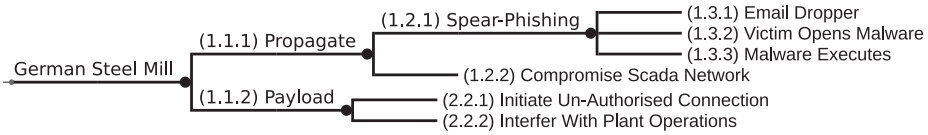

Figure 12: SAND model German steel mill.

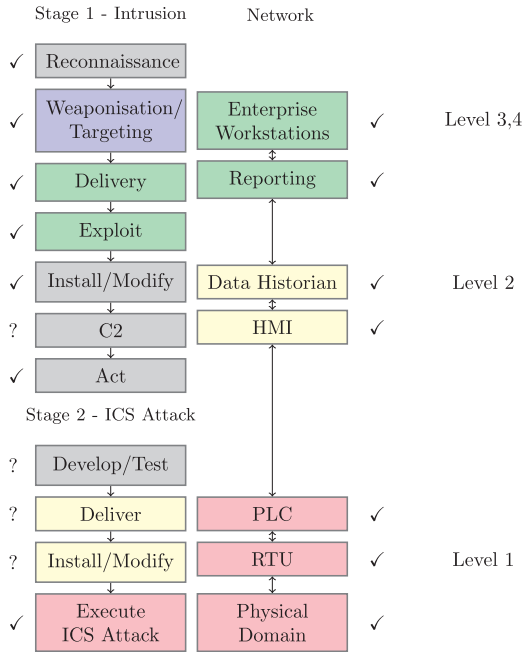

Figure 13: Mapping the German steel mill incident to the ICS-KC and ICS network.

attack, which includes the three primary steps that this attack is assumed to have performed; 1.2.1 Spear-Phishing followed by 1.2.2 compromising the SCADA network, and finally the intentional (or unintentional) payload that interfered with plant operations.

Figure 13 maps the ICS-KC to the attack and network. Stage 1 of the kill chain has been completed, but it is not possible to confirm whether C2 communication was established. In terms of the graphical representation, here the use of a question mark is introduced to indicate a lack of knowledge about whether a given Stage in the kill chain was executed. It is only possible to map the attack execution step with any certainly in Stage 2, as it is not clear how much testing was performed, or how the attack was delivered to the target.

Duqu2.0

Duqu2.0 is primarily used for information gathering and exfiltration. In 2014, Kaspersky [33 identified the re-emergence of the Duqu malware, which was first seen in 2011 [49]. Duqu, and by extension Duqu2.0, exhibit the same structure and design philosophy as Stuxnet, leading researchers to believe that the creators of Duqu had access to Stuxnet's source code. Kaspersky discovered Duqu2.0 active on their network, and it was believed they were a target. There is speculation that the malware was used to spy on delegates at Iranian nuclear talks [50, 51], as it was also found at a hotel where many of the attendees were staying.

Duqu2.0 has been modelled in Fig. 14. Note that a more in-depth SAND analysis can be found in ref. [52], which details how the malware itself operates, and further elaborates its espionage functions, which are not ICS-specific. Focusing on the first branch of the SAND representation (1.1.1 Propagate), it includes spear-phishing followed by lateral movement, as seen in many of the attacks previously discussed. However, Duqu2.0 is different in that it compromises the Windows domain controller, and then uses it to push the malware to other hosts (2.3.3). There are two main versions of the malware, 'full' and 'light', and both have the capability to reside only within RAM. The full version is around 18 MB and contains the complete set of features, while the light version contains the minimal feature set necessary to gain a foothold within a system and establish C2 communication, after which additional plugins are downloaded and installed as required. Another unique feature of Duqu2.0 is the C2 handler (1.2.3), which creates an internal proxy for the infected clients within the network. Data can be covertly transferred in and out of the network without raising suspicion using a number covert method (3.3.1, 3.3.2, and 3.3.3). While there are a number of different payload types, as discussed in refs. [52, 53], for this analysis, the payloads are generalized into reconnaissance and attack. Furthermore, reconnaissance (2.2.1) is split into network and system reconnaissance branches. Network reconnaissance (1.3.1) comprises enumeration of hosts and network shares, as well as detection of network sniffers. System reconnaissance (1.3.2) involves monitoring the host of the malware to collect all manner of host data such as connected devices, process information and remote connections. Additional features of the system branch include password stealing (2.4.7), which is used for additional lateral movement, by grabbing SSH keys, email, and web browser databases. Where Duqu2.0 differs from Duqu, is that it specifically attempts to locate files (2.4.8) pertaining to HMI operations, such as 'data.hmi' or 'val.dat', and files from the '/Int/HMI/' and '/LG/HM/' directories.

Figure 15 maps Duqu2.0 to the ICS-KC. In this attack we only have reporting of data collection and espionage, which amounts to the completion of Stage 1. Nonetheless, this provides the adversary with significant information about the target, along with the provision of a very capable C2 infrastructure that can persist due to the compromise of domain controllers and the malware's ability to reside wholly within RAM. However, it can be seen that this attack only targets network levels 3 and 4 for reconnaissance, and at this point in time it did not have supported features/plugins for interacting with levels 1 and 2.

Ukraine power outage 1 (2015)

The first successful attack on a power distribution system, enabled by deliberate interference with the process control, was in Ukraine on 23 December 2015. ICS-CERT [54] reported that three Ukrainian oblenergos (electrical distribution operators) experienced coordinated attacks within 30 min of one another, forcing them to switch from digital to manual operations. This resulted in a loss of power that affected thousands of people for several hours. Once the attackers had remotely disabled the power, they wiped the master boot record (MBR) of control systems to prevent the machines from booting, and then disabled the uninterruptible power supply (UPS) causing the systems to shut-down, thus preventing remote re-enabling of the power. At the same time a denial of service (DOS)

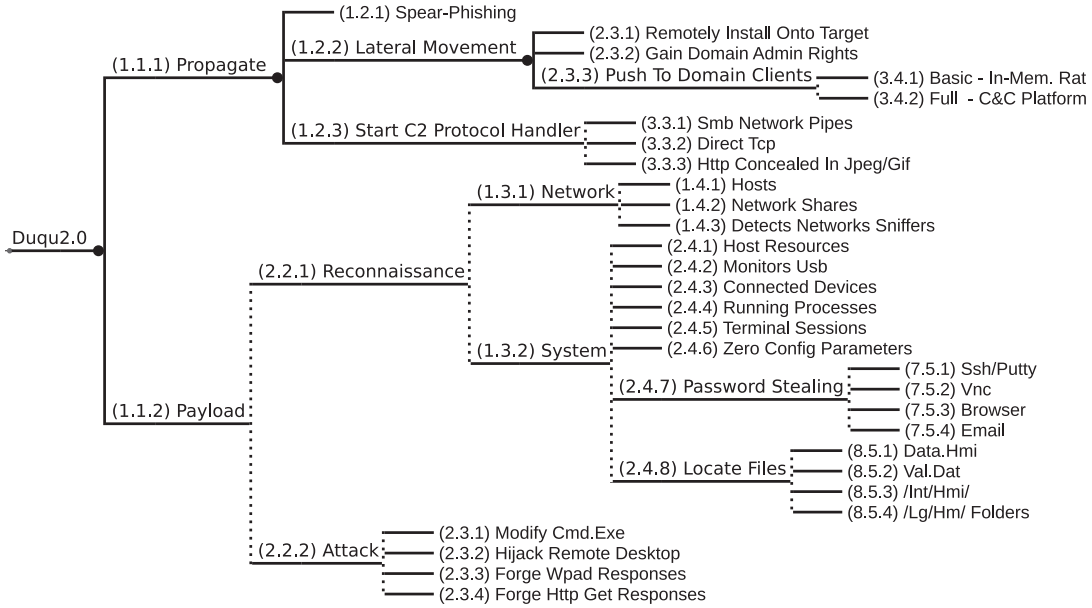

Figure 14: SAND model of Duqu2.0.

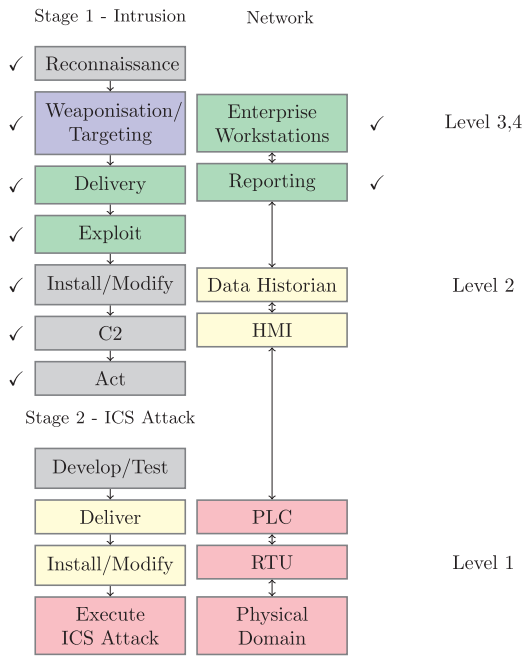

Figure 15: Mapping Duqu2.0 to the ICS-KC and ICS network.

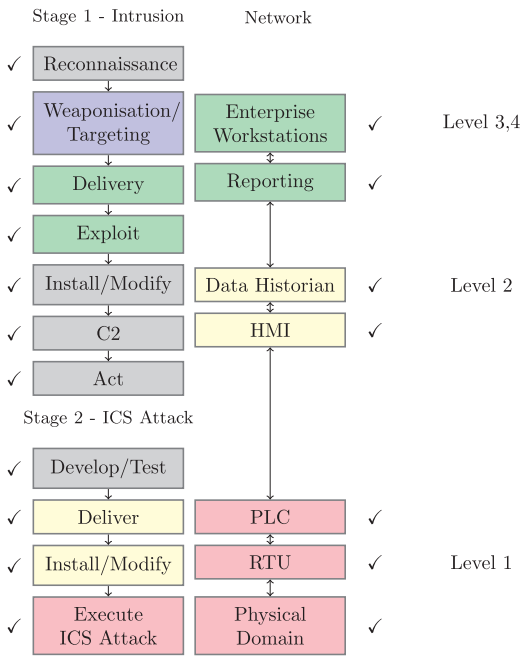

Figure 16: Mapping the Ukrainian Power Outage 1 to the ICS-KC and ICS network.

attack was performed that prevented telephone communication with on-site engineers.

Figure 16 shows how this attack maps to the full ICS-KC and how the adversaries were able to compromise the entire control network from business to process control networks, including the physical domain. Lee *et al.* [55] and Beach-Westmoreland *et al.* [56] have performed an analysis of the events that happened. It is believed that the attackers compromised the system via multiple methods: (i) spear-phishing; (ii) a variant of Black Energy; and (iii) taking advantage of a Microsoft Office macro vulnerability. This gave the attackers access to the business network, where they proceeded to establish C2 communications which allowed them to gather information about the environment and identify services. They were able to gain legitimate login credentials to remote access and systems which provided them with a discrete and more reliable foothold compared to the C2 methods, as well as access to the SCADA and Process Control Networks. These events align closely with Stage 1 of the ICS-KC. Once they were able to enumerate and identify devices in the process control enclave, they developed malicious firmware for the serial-to-Ethernet converters, which would cause them to become disabled once applied. They also identified which machines are critical to the operation of the network, and developed

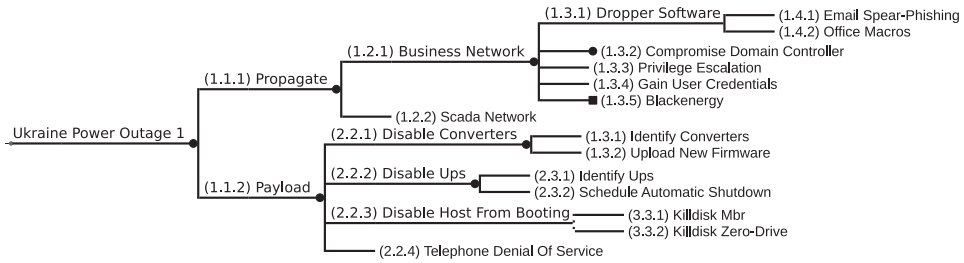

Figure 17: Ukraine power outage of 2015.

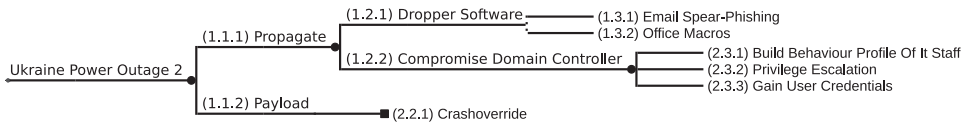

Figure 18: Ukraine power outage of 2016.

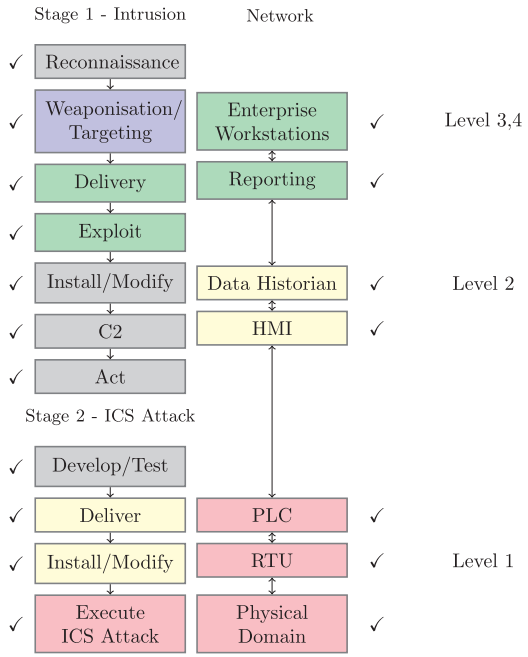

Figure 19: Mapping the Ukrainian Power Outage 2 to the ICS-KC and ICS network.

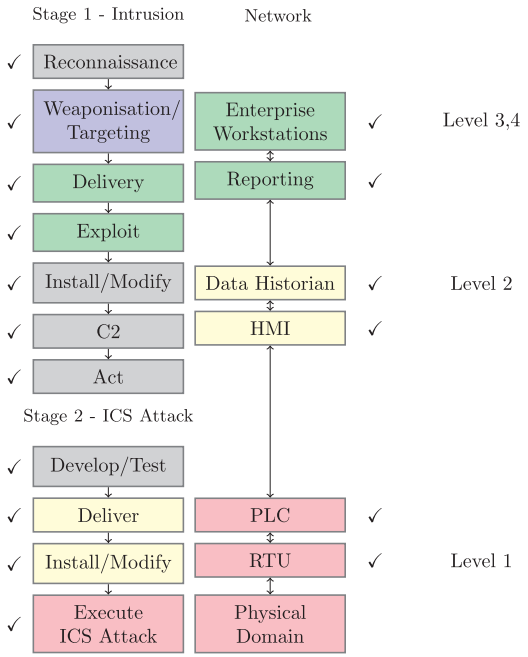

Figure 20: Mapping CrashOverride to the ICS-KC and ICS network.

a wiper application to disable them. At the same time, they scheduled the UPS system to shut-down after their attack was complete. Finally, they performed their ultimate attack objective which was to remotely open circuit breakers using the HMI, after disabling local access to the machine for the operators.

Figure 17 shows the SAND model derived for the 2015 Ukraine power outage. The propagate branch (1.1.1) has two conditions, business network (1.2.1) followed by SCADA (1.2.2). The initial dropper (1.3.1) is shown, followed by a privilege escalation on the domain controller to gain user credentials for administrator and VPN accounts (1.3.2). This is followed by the execution of the BlackEngery malware (1.3.3). The next branch is payload (1.1.2), which consists of disabling the serial-to-Ethernet converters (2.2.1), via a malicious firmware update. Backup power (2.2.2) is disabled by scheduling automatic shutdown. Finally, the HMI hosts are disabled via the KillDisk malware (2.2.3). This is one of the most complex, and successful cyber-attacks on critical infrastructure. It is worth noting that although it caused significant disruptions to physical operations, and can be mapped to all steps in the ICS-KC, each of the individual elements of the attack, as shown in Fig. 17, were not themselves exceptionally sophisticated.

Ukraine power outage 2 (2016)

Exactly 1 year later, Ukraine experienced another attack on their power systems [57], along with other services such as Railway, Pension Funds of Ukraine, Treasury of Ukraine, Ministry of Finance and Ministry of Infrastructure. In many of the intrusions, the adversary had compromised the networks approximately six months before the attacks took place. This analysis will focus on events relating to a loss of electrical power supply for around one hour on 17 December 2016. Figure 18 shows the attack tree for this analysis. The payload branch is a JOIN to the CrashOverride analysis that will be discussed further in the section. Figure 19 presents the ICS-

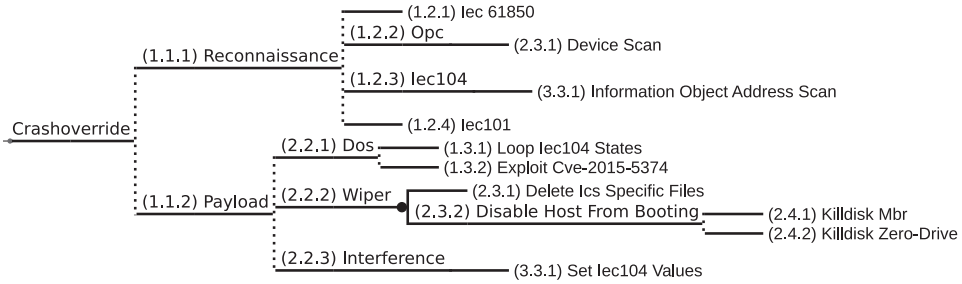

Figure 21: SAND model for CrashOverride.

Table 1: The two stages of the ICS-KC

(a) Stage 1: Cyber intrusion preparation and execution	
1. Planning	Reconnaissance
2. Preparation	Weaponization
	Targeting
3. Cyber intrusion	Delivery
	Exploit
	Install/Modify
4. Management and enablement	C2
5. Sustainment, entrenchment, development, and execution	Action
(b) Stage 2: ICS attack development and execution	
6. Attack development and tuning	Develop
7. Validation	Test
8. ICS attack	Deliver
	Install/Modify
	Execute ICS attack

Table 2: File extensions targeted by the data wiper module of CrashOverride from ref. [15]

File extension	Usage
.pcmp	PM600 project (ABB)
.pcmi	PCM600 IEC file (ABB)
.pcmt	PCM600 template IED file
.CIN	ABB MicroScada
.PL	Programmable logic file
.paf	PLC archive file
.SCL	Substation configuration language
.cid	Configured IED description
.scd	Substation configuration description

KC, with all the steps ticked, indicating that the whole network is compromised.

Spear-phishing and Microsoft Office macros were again used for the initial compromise (1.2.1), but compared to the previous attack in 2015 there was a significant increase in complexity. KillDisk was also used again to zero out the MBR, as well as zeroing all the files on the disk if it is not rebooted. This includes all the logs and actions the adversary may have performed. In some cases, it was found that 60% of the malware code was apparently present only to cause

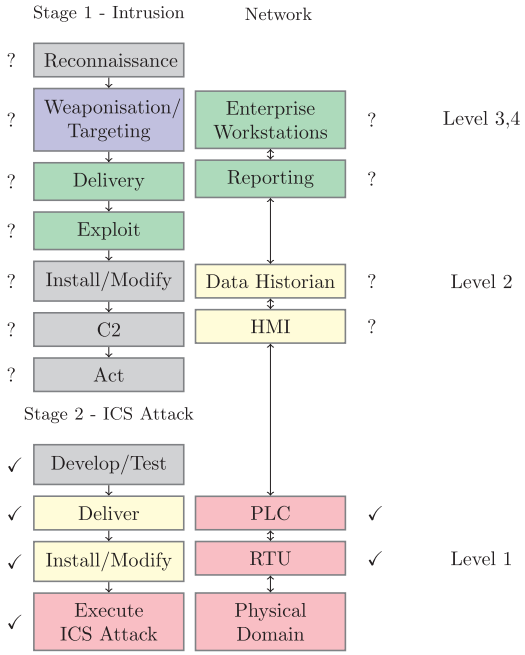

Figure 22: Mapping TRISIS to the ICS-KC and ICS network.

confusion. Once the attackers had access to the network they quickly attempted to gain access to core systems such as domain controllers [58], which the attackers could compromise and use to extract user credentials (1.2.2). The attackers performed their activities without detection, using normal system administrator tools built into the operating system, thus disguising their actions. In such cases, standard software has a few advantages over malware: (i) it is already available, so does not need to be developed and deployed; and (ii) attackers can mask their activities, by mimicking normal staff activities and performing actions during working hours to avoid suspicion. They were able to remain undetected by building a behaviour profile of IT staff by analysing the service and application logs, and familiarizing themselves with the programs being used. They compromised the domain controller and took advantage of known vulnerabilities to gather admin account passwords, such a plaintext password of logged-on users stored in memory. Using Mimicatz[6] on a remote terminal access sever, it is possible to get hundreds or thousands of user passwords, depending on the configuration. Finally, and an important element of this attack, is that when they performed the power cut it was automated without

6 https://github.com/gentilkiwi/mimikatz - A tool to extract plaintext passwords, hash, PIN code and kerberos tickets from memory of a Microsoft machine.

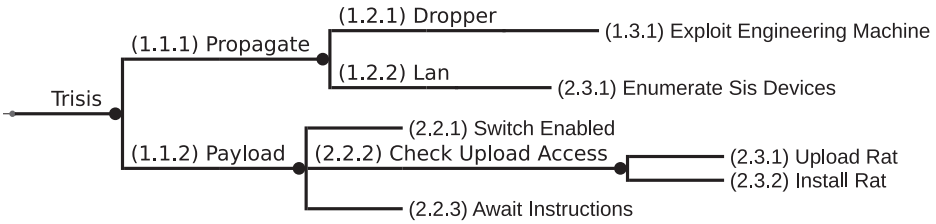

Figure 23: SAND model for TRISIS.

needing to hijack the operator's HMI. This showed a significant increase in sophistication and domain expertise compared to the previous attacks in 2015.

Crashoverride

CrashOverride is a complex modular piece of malware designed to target ICS devices. The anti-virus company [59] first detected the malware on 8 June 2017. They collaborated with Dragos, Inc., an ICS security company, and jointly published an analysis [1]. Dragos confirmed that CrashOverride was used in the Ukrainian power outages of 2016. A month later the ICS-CERT released an alert [60], reiterating the two reports. Figure 20 shows that CrashOverride maps to the complete ICS-KC, and can directly affect the whole process control enclave. As this analysis is limited to the malware framework (rather than the whole Ukraine attack), the final Stages of the ICS-KC are not marked.

Figure 21 shows the SAND model derived for the CrashOverride attack. There are two main branches, 1.1.1 Reconnaissance and 1.1.2 Payload. The reconnaissance branch consists four fieldbus protocols IEC61850, OPC and IEC104/101. Each of these protocols is used by CrashOverride to interact with ICS devices and servers. The IEC61850 (1.2.1) module can be used either by specifying a configuration file, or enumerating the network for devices that respond to certain IEC61850 commands. Similar to Havex, it uses OPC to enumerate (1.2.2) all OPC servers whilst looking for a string related to ABB software (a common ICS equipment vendor). If it finds the string matching ABB it will write 0×01 twice, which would overwrite the original value. 0×01 is outside the specified values of the device, with the vendor stating that the resulting effects are unknown. The IEC104 (1.2.3) module is a fully functional implementation of the protocol, allowing the malware to configure itself as the MASTER controller, and start communicating with any IEC104 SLAVE devices on the network.

The Payload branch (1.1.2) contains three leaf nodes. The first is DOS, and CrashOverride supports two methods for performing a DOS attack. Method 1 is the IEC104/101 module (1.3.1) which can set values, enumerate Information Object Addresses (IOAs), continuously set the IOA to open or toggle between states. Method 2 is a DOS attack, which exploits CVE-2015-5374 (1.3.2) relating to the SIPROTEC digital relay, causing it to enter a non-responsive state by sending a maliciously crafted packet. This can only be resolved by flashing the device. The second payload leaf is the wiper (2.2.2) function that deletes files on the host machine causing it to fail to boot. Interestingly, it also searches for a list of ICS configuration files across the local hard drive and network drives which it proceeds to delete. It targets files that are unique to ABB's PCM600 device. Table 2 provides a list of file extensions that the wiper module targets, which include a wide range of ICS-specific file types that are important for stable ICS operation. Finally, the third payload leaf node, Interference (2.2.3), acts upon two protocols IEC104 (if connected via Ethernet) and IEC 60870-5-101 (IEC101) (if connected via serial). Here, it is possible to attack a SLAVE device by sending a sequence of on/off values to the IOA.

TRISIS

TRISIS is named because it targets a Schneider Electric Triconex SIS device (although other names include HatMan and TRITON). A SIS is used as a final safety system, designed to shut-down a process before physical harm is caused. They are designed to be highly dependable and functionally robust when compared to a normal PLC. They conform to safety integrity level standards such as IEC61508, which enforce security measures such as error correcting memories and redundant components that are designed to fail in an operationally safe way, rather than continue operating in a dangerous manner. In most cases, when a SIS is deployed it is for a specific use-case and is unique to that specific deployment. TRISIS is a Remote Access Trojan (RAT), and was discovered when an operator in the Middle East noticed an issue with their site which triggered a routine internal investigation into the cause of a system failure. From there, the site engineers discovered that code had been uploaded to the Triconex SIS outside the scheduled maintenance window. They called in FireEye[7] to perform a forensic investigation of the engineering station and the SIS device which caused the shutdown. The leading technical analyses of TRISIS are Johnson *et al.* [61] and Dragos [2], while ICS-CERT [62] also released an analysis of the malware which details the worst case scenarios based on the features identified from analysis of the malware. Figure 22 provides a mapping of the TRISIS malware against the ICS-KC, but there is not enough evidence to know precisely what happened at Stage 1, and how the attack reached the process control enclave. Question marks are therefore used to note this uncertainty in the diagram.

Figure 23 shows the SAND model of the TRISIS malware. The initial propagation step focused on the engineering station (1.2.1), where the Triconex SIS was configured without using the official software. This suggests the attackers may have reverse engineered the relevant Schneider protocol. One year before the malware was discovered [63], published an academic proof of concept showing reversing of the Triconex protocol and subsequent attacks which compromise the system. The TRISIS malware can perform a UDP broadcast to port 1502 to enumerate (2.3.1) Triconex controllers. Once a controller is identified it attempts to upload a dummy program (2.2.2), and if successful, TRISIS will build a malicious program and send it to the device (2.3.1–2). Once the malicious program is complete, it is overwritten with a dummy program to remove any trace of the malicious program. There are some requirements which need to be met before TRISIS can upload the program to the controller:

7 FireEye, Inc. is an incident response company.

1. access to SIS network, either physical or remote;
2. ability to load malware onto the TriStation Programming Terminal (in this case, it is not publicly known how the engineering station was compromised); and
3. the Tricon physical key switch must be in PROGRAM mode for the Triconx device to allow write access (2.2.1).

No one has yet identified the dropper within the malware, or any communication to C2 modules, which are features typically found in similar malware. This could imply that a cell-styled development and deployment was used to prevent identification of the intent, or it could have been an inside job. The objective of the malware is to install a working RAT, suggesting that the objective is not to interfere with operations or damage the controller, but rather to have control over it. The shellcode which is sent to the SIS runs in the scan cycle and performs a privilege escalation to get read/write access to the firmware. This was a zero-day exploit, but it has been designed in such a way that it will only work on a specific Triconex device, model and version. This highly targeted aspect is similar in nature to Stuxnet, which may point to state involvement. Once read/write access is gained it will run a process in memory which listens for specific debug commands sent over the network, then execute. TRISIS was caught before it was able to cause significant damage to the system, and there is speculation that another stage of the TRISIS payload is being developed.

Summary

Based on the decomposition of the attacks in the previous section, Table 3 below aims to synthesize the key points of each attack, highlighting the pertinent details, with a particular focus on the characteristics and trends seen across the adversaries, attack objectives, malware and expertise related to the process control domain.

Discussion

As a general observation, most of the attacks follow a pattern of: reconnaissance, research and exploit. Depending on the threat actor, there may be long periods of time between the research and attack stages. As seen with Stuxnet, Duqu2.0 and the Ukrainian power outages, the adversaries compromised the system for several months before executing the final attack. In these attacks the most likely reason is that the actors where gathering intelligence and developing the exploit(s) before completing their goals.

The presented analysis shows an evolution of the level of attack sophistication over time. This is especially clear from the attacks on Ukraine, where the attackers progressed from manually interacting with the HMI using common malware, to a fully automated attack with custom built malware, within the space of a year. This is likely due to the adversary's increase in domain expertise and skill. Another trend is the re-purposing of malware, as seen in BlackEnergy and Duqu2.0. Both started as generic data exfiltration and DOS platforms, but evolved to target specific ICS files and services. This evolution, or re-purposing, is something that is likely to increase as we start to see more malware frameworks being developed. CrashOverride is a good illustration of this potential trend, as it provides adversaries with the tools necessary to gain a foothold, and a platform to swap in or out modules that are applicable for specific target sites, technologies and devices.

The attacks that interacted with industrial devices did so in a more covert manner than is typically seen with everyday malware. Assante [64] proposed that TRISIS is only part of a larger attack, and that there is another missing stage yet to be discovered. Since TRISIS targeted a SIS, which is harder to compromise than the HMI or other SCADA systems, it is believed this stage of TRISIS was designed to amplify an attack on the SCADA systems, which were easier to compromise. As TRISIS was the first known attack on a SIS component, this aligns with a trend of attacks evolving, becoming more customized, and even that attackers are returning to compromised systems having developed new capabilities for greater impact. Finally, a related trend that seems likely to continue is behaviour profiling, accompanied by a long reconnaissance period. This was seen in the second Ukrainian attack where the adversaries, after gaining an unauthorized foothold, quickly performed privilege escalation, profiled the behaviour of system operators and then mimicked their actions. This significantly reduces the risk of an adversary being detected, unless the defenders are actively monitoring key systems and looking for relevant indicators of compromise.

Derived commonalities

The objective of deriving commonalities is to provide data based on contemporary intrusions that can be used to improve detection and mitigation strategies against future attacks on infrastructure. Figure 24 shows a SAND model of commonalities derived from the nine presented attacks. There are three main branches: Reconnaissance, Propagate, and Payload. Mapping this model to the ICS-KC, both reconnaissance and propagate are within the first stage, while the payload fits within the second. An alternative terminology might have been to categorize each attack into their hypothesized intention, for example, espionage, destruction, etc.; however, due to the nature of the attacks we lack solid data about their true intentions, and can only base the conclusions on what has been observed. At a high level, we classify the resulting intrusion into Reconnaissance or Destruction. Havex, Duqu2.0, BlackEngery and the speculation around the German Steel Mill incident are attacks which are believed to have comprised reconnaissance and data exfiltration only. The other attacks belong to the destruction class, with each of them causing some form of destructive activity before being detected. Nonetheless, they also performed reconnaissance at the start of the attack and during continued activities.

The first branch, Reconnaissance (1.1.1), may be performed before or after intrusion, depending on the information the adversary is attempting to gain. The adversary may observe tools, time of day, and how members of staff go about their day-to-day activities (1.2.1, 1.2.2), this will allow the adversary to hide in plain sight. They may also monitor network traffic (1.2.3, 1.2.4) in order identify their primary target, or additional targets for lateral movement. While on the host, the attacker will again monitor running processes (1.2.5) to find exploits or to gain additional information about the environment. The second branch, Propagate (1.2.1) consists of two leaf nodes, Lateral Movement, and C2. These have been separated because adversaries have been seen to perform one and not the other. Lateral Movement (2.2.1) contains an interesting trend called 'Living off the land' (1.3.1), which means that the attacker uses the tooling already loaded on the victim's machines. Not only is this easier to operate, but it allows them operate without drawing additional attention. This is in part due to the tightening up of security, and increased detection methods being deployed. Gaining user permission can be performed by cracking the domain controller, to get access to users' passwords, or using stolen credentials (1.3.2, 1.3.3). Stolen credentials may also be gained from the initial reconnaissance branch. The most common methods used for intrusion were spearphishing, removable storage, and water hole attacks (1.3.4, 1.3.5,

Table 3: Key take away from each of the nine attacks

Attack	Key points of attack
Stuxnet	Adversary remained hidden from detection for long period of time, gaining a great deal of insight into the target
	Adversary developed complex malware, never before seen
	Malware used a combination of zero-day exploits
	Adversary knew exactly what system properties to manipulate to achieve their target objective
	Adversary showed significant knowledge of industrial equipment and the system processes
	Malware used methods to disguise actions to appear legitimate
	Attack resulted in physical damage
	Attack was targeted, so the impact was limited to one specific location and device
	Malware had a built-in sunset date to shut it down and prevent it spreading
Havex	Malware bypassed security measures by masquerading as a legitimate update, and penetrated directly to the SCADA enclave
	Malware performed OPC network enumeration, showing ICS expertise
	This has the added benefit of disguising the scan within normal field bus protocols
	Attack had no known physical impact
	Adversary appears to have limited their objectives to espionage and data exfiltration
BlackEnergy	Malware is complex and shows that a development life cycle has been used with each version gaining more features
	Adversary adapted to changes in systems and improved methods from version to version
	Adversary demonstrated a basic level of domain expertise, and had identified a number of known SCADA vulnerabilities and performed reconnaissance
	Adversary objectives appear to be espionage and data exfiltration
	Adversary/malware did not target specific ICS equipment, and no known physical damage
	Malware was general purpose, which can run on a large variety of hosts
German steel mill	Adversary/malware successfully performed lateral movement without detection
	Adversary/malware compromised both the business enclave and the process control enclave
	Adversary/malware triggered an unintentional shutdown of the plant, possibly triggered by a SIS
	The only evidence of impact reported is monetary cost to the company
Duqu2.0	Malware is highly complex code including three zero-day exploits
	Malware uses modular framework allowing for deployment on different operations without recompilation
	Malware uses complex methods of data exfiltration
	Adversary showed a proficient level of knowledge of HMIs and knowledge of the files pertaining to ICS operations
	Malware designed to remain undetected for long periods of time
	Malware designed to operate on any system taking advantage of many known/unknown vulnerabilities
	No physical damage has been attributed, but malware could easily be re-purposed to attack ICSs
Ukraine Power Outage 1	Adversary developed specialized malicious firmware for serial-to-Ethernet converters
	Adversary maintained covert operations for prolonged period of time
	Performed highly synchronized attack spanning multiple stages
	Adversary demonstrated comprehensive knowledge of operational equipment needed to perform the attack
	Malware wiper used to hinder the restoration of the network, by disabling the remote access
	Adversary performing a telephone DoS shows they considered the incident response plan of the target
	Attack caused irreparable damage to serial-to-Ethernet converters
	Attack erased MBR of control computers, preventing boot
	Attack leveraged scheduled service outage of UPS
	Attack ultimately caused de-energization of parts of the power network
Ukraine Power Outage 2	Adversary remained undetected for a long period of time by mimicking staff behaviour
	Malware used was sophisticated, along with automated attacks
	Adversary demonstrated comprehensive knowledge of operational equipment needed to perform the attack
	Adversary demonstrated an increase in ICS knowledge to allow for automated shutdown
	Compared to the 2015 attack the impact of this attack is not as severe, with only an estimated 1 h loss of power
	However, the malware can easily be applied to other ICS using similar protocols.
CrashOverride	Malware used modular framework allowing multiple teams to separately develop specific capabilities
	Malware demonstrates a large number of features which can be used for both reconnaissance and attack
	Malware supports communication with a number of widely used field bus protocols
	Malware support both read and write commands that can be executed to field bus devices
	Identified in a site in the middle east
	This framework was attributed to the second Ukrainian power outage in 2016
	Potential for this framework to be used elsewhere is very high
TRISIS	Adversary appears to have reverse engineered the Triconx protocol and SIS device
	Adversary developed complex shellcode and in-memory-only remote access trojan
	Adversary/malware targeted a specific ICS device, model and version
	Adversary needed to have access to hardware and required a significant investment of resources for attack development
	Attack triggered a shutdown of physical systems
	Attack is unique to this specific model

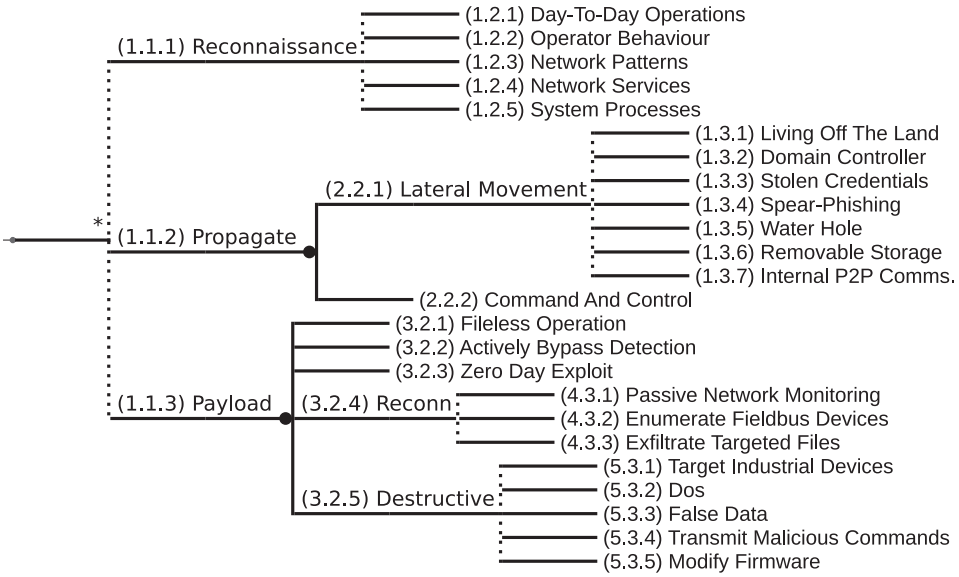

Figure 24: SAND model of commonalities.

Table 4: Attack commonalities and their typical consequences

Commonality	Consequence
Internet enabled	Possible to exfiltrate data out of the network through internet enabled hosts
	Allows downloading of dropper software and C2 communication
Network enumeration	Mapping hosts on the network for future attacks. Provides attackers with intelligence about the network
Unauthorized network access	Allow lateral movement
	Download additional capabilities and upload confidential data
Fieldbus enumeration	Mapping the industrial process, providing the attacker with confirmation of target or allows them to develop targeted malware for the process
Compromised host machine	Provides attackers with intelligence about the system. They can build profiles about IT staff, and identify patterns which can be used to further embed themselves into the network
	Asset identification through log analysis and network monitoring
Falsified data	Loss of trust of the network/device/software
	Incorrect data reported, physical damage
Unauthorized binary execution	Loss of trust in host
	Allows additional capabilities and allow an adversary gain a foothold
Credential harvesting	Provides adversaries with authentic, disguised access to services
Malicious firmware	Falsified and unreliable readings returned to operator, or actions applied to physical system
	Physical damage/Loss of life

1.3.6). Internal peer to peer communication has been used by malware, such as BlackEnery and Duqu, to provide compromised hosts with additional instructions, and to exfiltrate data.

In the third branch, Payload (1.1.3), we observed an increasing trend towards complex malware that resides only in memory, called Fileless Operation. This was seen in Duqu and CrashOverride. Many of the recent attacks took advantage of zero-day exploits and sophisticated mechanisms to bypass detection systems. Since many of the attacks were multi-function, it makes sense to include a branch for Reconnaissance (Reconn, 3.2.4) and Destructive (3.2.5) operations related to the malware payload. The Reconn branch includes operations comprising monitoring of network traffic, enumerating fieldbus devices, and functions to exfiltrate files. The Destructive branch (3.2.5) mainly relates to interacting with fieldbus devices, and has gained prominence in attacks since 2017, whereby malware is now observed to be supporting DOS, data injection, and firmware update functions for fieldbus devices. This branch increasingly is seen to provide adversaries with an automated way to interact with critical process control devices.

Distilled set of commonalities

Based on further analysis across the nine SAND models and the SAND model of commonalities derived above, this section introduces a distilled set of commonalities, selecting features that are most common throughout the observed attacks and likely to cause a high level of impact. Table 4 captures each of the distilled commonalities, and highlights the consequence of each commonality.

Table 5: Attack commonalities and association with the nine attacks

Commonalities	Stuxnet	Havex	Black Energy	German Steel Mill	Duqu2.0	Ukraine power Outage 1	Ukraine power Outage 2	Crash Override	TRISIS
Internet enabled	×	√	√	√	√	√	√	√	?
Network enumeration	√	√	√	?	√	√	√	√	?
Unauthorised network access	√	√	√	?	√	√	√	√	√
Fieldbus enumeration	√	√	×	?	×	√	√	√	√
Compromised host machine	√	√	√	√	√	√	√	√	√
Falsified data	√	×	×	?	×	√	√	√	×
Unauthorised binary execution	√	√	√	?	√	√	√	√	√
Credential harvesting	×	×	√	√	×	√	√	√	×
Malicious firmware	√	×	×	?	×	√	√	√	√

Table 5 lists nine attack commonalities derived from the analysis. For the German Steel Mill and TRISIS, note that a '?' has been placed where there is insufficient evidence in the literature to confirm the presence of a number of commonalities.

Network-enabled devices, whether they are bidirectional or unidirectional, provide an adversary with an attack surface. Internet connected devices allow remote interaction, data exfiltration and downloading of additional software, either by using compromised credentials, as in the second Ukrainian attack, by using C2 via RATs, or other communication methods such as used by Duqu2.0. Each of the analysed attacks manipulated Internet-connected device, with the exception of Stuxnet, which was capable of performing highly automated actions towards its goals.

Network enumeration has been seen in both the business and SCADA enclaves, and provides the adversary with detailed information about the network, and services that are operating. This stage is often contributing to the reconnaissance and development stages of future attacks. Network enumeration has been observed being performed both in a simplistic way, using common network scanning techniques, and in a targeted manner via specific fieldbus scanning. The Havex malware was the first of the reported attacks to scan the network in a targeted manner, in this case for OPC devices. More recently, in comparison, Crashoverride supported a large number of fieldbus protocols for both IP and serial links, demonstrating an increase in the sophistication of adversary behaviours.

Unauthorized network access is the only common element throughout all of the attacks. It is typically performed once the adversary gains a foothold onto the system. Unauthorized network access consists of interfacing with other networked devices, either by unusual means or following well known norms, with the resulting action to gain additional privileges or reconnaissance information about a target. A significant step can be the fingerprinting of devices once they are discovered. For example, Stuxnet and TRISIS identified target devices based on the results of unauthorized network scanning.

A fully compromised host on the network can provide an attacker with much more information than can be gained from the network alone. The type of information naturally depends on the host compromised and the types of services it operates. By performing log and file analysis of a compromised host, it is possible to identify additional devices and information, along with manuals and documents disclosing information about specific systems at the target site. For example, in the case of the Ukrainian attacks, the adversaries built up a profile of user behaviours based on access logs, which enabled them to better hide their activities.

Binary execution can be categorized as legitimate (e.g. an authorized system binary) or illegitimate (e.g. a malware payload or exploit). Although both types of binary are used throughout the attacks, their impact on each system varies. Performing application whitelisting will prevent illegitimate applications, such as in Havex, Crashoverride and TRISIS attacks, from executing and delivering their payloads. This would not stop adversaries from misusing existing binaries to advance their goals, such as in the first Ukrainian power outage, which used remote desktop as a vector to eventually trip the circuit breakers.

In many of the attacks, once user credentials were gained, it allowed the attacker unlimited lateral movement, and bypassed network segmentation and separation protections. There are two credential harvesting types, passive and active. Passive means using default or well-known passwords that can be found by using open source intelligence such as search engines and manuals. Active means exploiting a target system to directly gain user credentials. On a Windows machine this is commonly done using mimikatz to extract plaintext passwords, hash, PIN code and kerberos tickets from the memory of a running machine. It can also perform pass-the-hash, pass-the-ticket or build Golden Tickets, similar to Duqu and Black Energy.

Compromising an embedded device requires the identification of an unknown vulnerability in installed firmware, or flashing a device with a vulnerable version. In the case of TRISIS, the Ukrainian attacks, and Stuxnet, the adversary pushed new firmware to devices to cause a loss of trust in them, and allowed the adversary to falsify data to the operators. In the Havex attack, the adversary compromised vendor support websites and served malicious firmware to the unknowing operators.

The commonalities models revealed that these highly targeted intrusions could have been detected and mitigated using existing methods, such as whitelisting binary execution (this is built into the Linux kernel since 2000 SELinux[8]). Finally, it is notable that some of the most exploited vulnerabilities were due to a lack of appropriate security practices, insufficiently customized to protect process control networks, and in particular a lack of basic measures such as passive network security monitoring. This may have prevented the adversaries ability to move laterally once inside the network as well as maintain C2 communication. This is unexpected, and could perhaps be overlooked in the context of zero-day exploits being used in many cases. Although unforeseeable, the zero-days facilitated

8 SELinux https://selinuxproject.org/

subsequent adversarial actions that themselves could have been discovered with improved network monitoring.

Conclusion and future direction

This article surveyed the landscape of contemporary attacks on critical infrastructure control systems, and provides an in-depth analysis of each attack based on publicly available reporting of each incident. Although most previous literature in this area focused on formally modelling single intrusions, we have taken a more holistic approach towards deriving a set of commonalities from an analysis across a comprehensive range of recent attacks.

How does this help detect or protect against future attacks? We believe that a number of future directions are made feasible by building on the presented analysis of attacks:

- it may be possible to expand upon the SAND models to allow quantification of the attack steps. By calculating which paths an attacker might take, and assigning it a monetary or skill value, we can identify which areas of an ICS network can be improved to mitigate such attacks in the future;
- the commonalities that have been proposed can be used to aid in detecting attacks by providing additional high-level IOC and knowledge about attacker tradecraft; and
- researchers developing intrusion detection systems can build attack verification use-cases and strategies based on the identified broad commonalities across all the analysed attacks. Furthermore, researchers may wish to adopt and replicate individual attacks, or branches of individual attacks, based on the detailed SAND models presented for each attack.

Acknowledgements

The authors wish to thank the anonymous reviewers for their valued feedback and time, in reviewing this article. We would also like to thank Dr Peter Popov and Professor Andrew Blyth for their feedback on this work.

Conflict of Interest Statement

None declared.

References

1. Dragos. *CRASHOVERRIDE—Analysis of the Threat to Electric Grid Operations*. Technical report. Dragos, Inc, 2017a.
2. Dragos. *TRISIS—Analyzing Safety System Targeted Malware*. Technical report. 2017b.
3. ICS-CERT. *Incident Response Summary Report (2009–2011)*. Technical report. 2012.
4. Grooby S, Dargahi T, Dehghantanha A. Protecting IoT and ICS platforms against advanced persistent threat actors: analysis of APT1, silent chollima and molerats. In: Dehghantanha A. and Choo K-KR (eds), *Handbook of Big Data and IoT Security*. Cham: Springer International Publishing, **2019**, 225–55.
5. Robinson M. The SCADA threat landscape. In: *International Symposium for ICS & SCADA Cyber Security Research*, 2013. UK: BCS.
6. Kovacs E. *Mining Industry Increasingly Targeted by Threat Actors—SecurityWeek.Com*. 2016. http://www.securityweek.com/mining-industry-increasingly-targeted-threat-actors. Accessed 2017-07-10.
7. Kriaa S, Bouissou M, Pietre-Cambacedes L. Modeling the Stuxnet attack with BDMP: towards more formal risk assessments. In: *Proceedings of the 2012 7th International Conference on Risk and Security of Internet and Systems (CRiSIS)*. 2012, 1–8.
8. Langner R. Stuxnet: dissecting a Cyberwarfare Weapon. *IEEE Secur Privacy* 2011b;**9**:49–51.
9. Assante M, Lee R. *The Industrial Control System Cyber Kill Chain*. Technical report. SANS Institute, 2015.
10. Jhawar R, Kordy B, Mauw S, *et al.* Attack trees with sequential conjunction. In: Federrath H. and Gollmann D. (eds) *ICT Systems Security and Privacy Protection. Number 455 in IFIP Advances in Information and Communication Technology*. Springer International Publishing, 2015, 339–53.
11. Schneier B. *Attack Trees*. 1999. http://www.drdobbs.com/attack-trees/184411129. Accessed 2017-07-10.
12. Popov P. Models of reliability of fault-tolerant software under cyber-attacks. In: *IEEE 28th International Symposium on Software Reliability Engineering*. 2017, 228–39. City, University of London Institutional Repository.
13. CPNI. *Firewall Deployment for SCADA and Process Control Networks*. Technical report. Centre for the Protection of National Infrastructure, 2005.
14. Stouffer K, Pillitteri V, Lightman S, *et al. Guide to Industrial Control Systems (ICS) Security*. Technical report NIST SP 800-82r2. National Institute of Standards and Technology, 2015.
15. Langner R. *Robust Control System Networks*. New York: Momentum Press (2011a).
16. Knapp E. *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems*. Waltham, MA: Syngress (2011).
17. Jovanovic P, Neves S. Dumb Crypto in Smart Grids: Practical Cryptanalysis of the Open Smart Grid Protocol. *IACR Cryptol* 2015;**428**:2015.
18. Jardine W, Frey S, Green B, *et al.* SENAMI: selective non-invasive active monitoring for ICS intrusion detection. In: *2nd ACM Workshop on Cyber-Physical Systems Security and Privacy*. ACM, 2016.
19. Luchs M, Doerr C. Last line of defense: a novel IDS approach against advanced threats in industrial control systems. In: Polychronakis M and Meier M (eds) *International Conference on Detection of Intrusions and Malware & Vulnerability Assessment*. Cham: Springer (2017).
20. Miller B, Rowe D. A survey of SCADA and critical infrastructure incidents. In: *Proceedings of the 1st Annual Conference on Research in Information Technology*. RIIT'12. New York, NY: ACM (2012), 51–6.
21. McLaughlin S, Konstantinou C, Wang X, *et al.* The cybersecurity landscape in industrial control systems. *Proc IEEE* 2016;**104**:1039–1057.
22. Lamp J, Rubio-Medrano CE, Zhao Z, *et al.* OntoEDS: protecting energy delivery systems by collaboratively analyzing security requirements. In: *Proceedings of the 2017 IEEE 3rd International Conference on Collaboration and Internet Computing (*CIC). 2017, 1–10.
23. Khan R, McLaughlin K, Laverty D, *et al.* STRIDE-based threat modeling for cyber-physical systems. In: *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, 2017, 1–6.
24. Ani UPD, He HM, Tiwari A. Review of cybersecurity issues in industrial critical infrastructure: manufacturing in perspective. *J Cyber Secur Technol* 2017;**1**:32–74.
25. Bhamare D, Zolanvari M, Erbad A, *et al.* Cybersecurity for industrial control systems: a survey. *Comput Secur* 2020;**89**:101677.
26. Zhu B, Joseph A, Sastry S. A taxonomy of cyber attacks on SCADA systems. In: *International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing*. Dayton: Air Force Institute of Technology (2011), 380–388.
27. Mateski M, Trevino CM, Veitch CK, *et al. Cyber Threat Metrics*. Technical report. Sandia National Laboratories, 2012.
28. Kotheimer J, O'Meara K, Shick D. *Using Honeynets and the Diamond Model for ICS Threat Analysis*. Technical, Software Engineering Institute, Carnegie Mellon University, 2016.
29. Ross R. *Security and Privacy Controls for Federal Information Systems and Organizations*. Technical report NIST SP 800-53r4. National Institute of Standards and Technology, 2013.
30. Byres E, Franz M, Miller D. The use of attack trees in assessing vulnerabilities in SCADA systems. In: *Conference International Infrastructure Survivability Workshop*. Libson: IEEE (2004).

31. Arnold F, Guck D, Kumar R, *et al.* Sequential and parallel attack tree modelling. In: Koornneef F. and van Gulijk C. (eds) *Computer Safety, Reliability, and Security, Number 9338 in Lecture Notes in Computer Science*. Springer International Publishing, 2015, 291–99.
32. Arnold F, Hermanns H, Pulungan R, *et al.* Time-dependent analysis of attacks. In: *Principles of Security and Trust (POST), Lecture Notes in Computer Science*. Vol. 8414. Berlin: Springer, 2014.
33. Sommestad T, Ekstedt M, Holm H. The cyber security modeling language: a tool for assessing the vulnerability of enterprise system architectures. *IEEE Syst J* 2013;**7**:363–373.
34. Sommestad T, Ekstedt M, Johnson P. A probabilistic relational model for security risk analysis. *Comput Security* 2010;**29**:659–679.
35. Holm H, Sommestad T, Ekstedt M, *et al.* CySeMoL: a tool for cyber security analysis of enterprises. In: *Proceedings of the 22nd International Conference and Exhibition on Electricity Distribution (CIRED 2013)*. IET (2013), 1–4.
36. Caltagirone S, Pendergast A, Betz C. *The Diamond Model of Intrusion Analysis*. Technical, Center for Cyber Intelligence Analysis and Threat Research Hanover. 2013.
37. Herr, Trey, PrEP: a framework for malware and cyber weapons (December 20, 2013). The Journal of Information Warfare, Vol.13, No.1, February 2014, Available at SSRN: https://ssrn.com/abstract=2343798 or 10.2139/ssrn.2343798.
38. Hentunen D, Tikkanen A. *Hackers Target Critical Infrastructure SCADA Systems with Havex Trojan*. 2014a. http://www.prweb.com/releases/2014-Havex-DHS-alert/SCADA-ICS-Hack/prweb11982848.htm. Accessed 2017-07-10.
39. Hentunen D, Tikkanen A. *Havex Hunts For ICS/SCADA Systems*. 2014b. https://archive.f-secure.com/weblog/archives/00002718.html. Accessed 2017-07-10
40. Lucian Constantin. *New Havex Malware Variants Target Industrial Control System and SCADA Users*. 2014. http://www.pcworld.com/article/2367240/new-havex-malware-variants-target-industrial-control-system-and-scada-users.html. Accessed 2017-07-10
41. Symantec. *Dragonfly: Western Energy Companies Under Sabotage Threat*. 2014. http://www.symantec.com/connect/blogs/dragonfly-western-energy-companies-under-sabotage-threat. Accessed 2017-07-10
42. Erik Hjelmvik. *Full Disclosure of Havex Trojans—NETRESEC Blog*. 2014. http://www.netresec.com/? page=Blog&month=2014-10&post=Full-Disclosure-of-Havex-Trojans. Accessed 2017-07-10
43. Harpes C. *Analysis of Havex*. 2014. https://malware.lu/articles/2014/09/03/analysis-of-havex.html. Accessed 2017-07-10
44. Rrushi JL, Farhangi H, Howey C, *et al.* A quantitative evaluation of the target selection of Havex ICS malware plugin. In: *ACSAC Industrial Control System Security*. Los Angeles, California (2015).
45. ICS-CERT. *Ongoing Sophisticated Malware Campaign Compromising ICS (Update B)*. 2014. https://ics-cert.us-cert.gov/alerts/ICS-ALERT-14-281-01B. Accessed 2017-07-10
46. Constantin L. Attack campaign infects industrial control systems with BlackEnergy malware. 2014. http://www.computerworld.com/article/2840164/attack-campaign-infects-industrial-control-systems-with-blackenergy-malware.html.
47. BSI. Die Lage Der IT-Sicherheit in Deutschland. 2014. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2014.pdf?__blob=publicationFile
48. Lee R, Assante M, Conway T. *German Steel Mill Attack*. Technical report. SANS Institute, 2014.
49. Bencsáth B, Pék G, Buttyán L, *et al.* Duqu: A Stuxnet-like malware found in the wild. *CrySyS Lab Tech Rep* 2011;**14**:1–60.
50. Schneier B. Duqu 2.0. 2015. https://www.schneier.com/blog/archives/2015/06/duqu_20.html. Accessed 2017-07-10
51. Zetter K. *Kaspersky Finds New Nation-State Attack—In Its Own Network*. 2015. http://www.wired.com/2015/06/kaspersky-finds-new-nation-state-attack-network/. Accessed 2017-07-10
52. Maynard P, McLaughlin K, Sezer S. Modelling Duqu 2.0 malware using attack trees with sequential conjunction. In: *Proceedings of the 2nd International Conference on Information Systems Security and Privacy*. SciTePress (2016), 465–472
53. Kaspersky. *The Mystery of Duqu 2.0: A Sophisticated Cyberespionage Actor Returns*. 2015. https://securelist.com/blog/research/70504/the-mystery-of-duqu-2-0-a-sophisticated-cyberespionage-actor-returns/. Accessed 2017-07-10
54. ICS-CERT. *Cyber-Attack Against Ukrainian Critical Infrastructure—Alert (IR-ALERT-H-16-056-01)*. 2016. https://ics-cert.us-cert.gov/alerts/IR-ALERT-H-16-056-01. Accessed 2017-07-10
55. Lee R, Assante M, Conway T. *Analysis of the Cyber Attack on the Ukrainian Power Grid*. Technical report. E-ISAC, 2016.
56. Beach-Westmoreland N, Styczynski J, Stables S. *When the Lights Went Out*. Technical Report. Booz Allen Hamilton, 2016.
57. Zetter, K. *The Ukrainian Power Grid Was Hacked Again*. 2017. https://motherboard.vice.com/en_us/article/bmvkn4/ukrainian-power-station-hacking-december-2016-report. Accessed 2017-07-10
58. Yasinskyi O, Sologub R. *APT Cyber Attacks in Ukraine*. 2017. https://issp.ua/news_article.php? id=204&l=en. Accessed 2017-07-10
59. ESET. *Industroyer: Biggest Threat to Industrial Control Systems Since Stuxnet*. Technical report. 2017.
60. ICS-CERT. CRASHOVERRIDE Malware ICS-ALERT-17-206-01. 2017. https://ics-cert.us-cert.gov/alerts/ICS-ALERT-17-206-01. Accessed 2017-07-10
61. Johnson B, Caban D, Krotofil M, *et al. Attackers Deploy New ICS Attack Framework TRITON and Cause Operational Disruption to Critical Infrastructure*. 2017. https://www.fireeye.com/blog/threat-research/2017/12/attackers-deploy-new-ics-attack-framework-triton.html. Accessed 2017-07-10
62. ICS-CERT. *MAR-17-352-01 HatMan—Safety System Targeted Malware*. Technical report. 2017.
63. Lim B, Chen D, An Y, *et al.* Attack induced common-mode failures on PLC-based safety system in a nuclear power plant: practical experience report. In: *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing*, Christchurch (2017), 205–10.
64. Assante M. *Triton/TriSIS—In Searchof its Twin*. 2018. https://ics.sans.org/blog/2018/01/29/tritontrisis-in-search-of-its-twin. Accessed 2017-07-10