SN Computer Science (2023) 4:318
https://doi.org/10.1007/s42979-023-01750-z

ORIGINAL RESEARCH

Estimating Time-To-Compromise for Industrial Control System Attack Techniques Through Vulnerability Data

Engla Rencelj Ling[1] · **Mathias Ekstedt**[1]

Received: 13 October 2022 / Accepted: 22 February 2023 / Published online: 8 April 2023

Abstract
When protecting the Industrial Control Systems against cyber attacks, it is important to have as much information as possible to allocate defensive resources properly. In this paper we estimate the Time-To-Compromise of different Industrial Control Systems attack techniques by MITRE ATT&CK. The Time-To-Compromise is estimated using an equation that takes into consideration the vulnerability data that exists for a specific asset and category of vulnerability. The vulnerability data is derived from an Industrial Control Systems specific vulnerability dataset. As a result, we present the mapping of the attack techniques to assets and categories of vulnerability, which makes it possible to apply specific vulnerabilities to the technique. We also present the method of how to estimate the Time-To-Compromise of the techniques and finally the values of Time-To-Compromise. After mapping the attack techniques to assets and category of vulnerability we are able to estimate the Time-To-Compromise and discuss its trustworthiness.

Keywords Time-To-Compromise · Cyber security · Industrial control systems · Attack techniques · Vulnerability data · MITRE ATT&CK

Introduction

Industrial Control Systems (ICS) are the backbone of our society. They are the systems that control our electricity and many other critical infrastructures. We are reliant on these systems and this makes them a target for cyber attacks by potential malicious actors. One of the most well-known examples of a cyber attack specifically targeting ICS is Triton. In 2017 an attack with Triton was able to shutdown a petrochemical facility [1]. The number of reported vulnerabilities for ICS are increasing [2] and it is important to create a method for assessing them. By assessing the Time-To-Compromise (TTC) of a vulnerability, we can estimate the time that it would take for an attacker to compromise an asset. Knowing this information can help us prioritising resource allocation to protect the system.

In this paper we apply our method for estimating the TTC for ICS, previously reported in [3]. We specifically extend that paper by applying the method to the ICS attack techniques found in the MITRE ATT&CK knowledge base [4]. The main contribution of this article is that the techniques of MITRE ATT&CK ICS can be assigned an estimated TTC. For example, the estimated TTC for the Man in the Middle technique used on a Human-Machine Interface (HMI) is 2501 days for a novice and 6 days for an expert hacker. The problem of how to estimate TTC is difficult because of the many variables involved. For instance, the type of attacker and how vulnerable the asset is will affect the TTC of the asset. The method for estimating TTC was first created by McQueen et al. [5] and our method for estimating TTC for the ICS domain, labeled TTC_{ICS}, is an adaption of that method. In our adaptation we, for example, use an ICS specific vulnerability dataset [6] to make the estimate ICS specific. In the next section we present the related work and then we give background to TTC, the ICS vulnerability dataset, as well as, the MITRE ATT&CK knowledge base.

This article is part of the topical collection "Advances on Information Systems Security and Privacy" guest edited by Steven Furnell and Paolo Mori.

✉ Engla Rencelj Ling
englal@kth.se

Mathias Ekstedt
mekstedt@kth.se

[1] Division of Network and Systems Engineering, KTH Royal Institute of Technology, Teknikringen 33, 100 44 Stockholm, Sweden

The continued sections includes the method, the results and finally the discussion.

Related Work

Several work exists that has developed the original TTC by McQueen et al. [5] further. In TTC_{ICS} [3] we update several parameters based on new research and make the estimate specific for the ICS domain. Nzoukou et al. [7] estimates the Mean-Time-To-Compromise (MTTC) for a whole system using a Bayesian network. They assign the individual MTTC values based on exploits instead of giving a more general MTTC for the components. Zieger et al. [8] developed β-TTC, which extends the original TTC and fixes some mathematical flaws. The mathematical flaw is for a variable that exists in the original TTC, but it does not exist in TTC_{ICS}. They also divide the vulnerabilities into confidentiality, integrity, availability and execution.

Both Nzoukou et al. and Zieger et al add the metric Common Vulnerability Scoring System (CVSS) [9] to their estimate to add the exploit complexity into the equation. Leversage and Byres [10] developed a Mean Time-To-Compromise (MTTC) interval where they considered frequency of reviews of the access control list (ACL) rules as part of the estimate. They also replace McQueen et al.'s fraction of vulnerabilities that are exploitable based on skill level with the concept skill indicator.

In this paper we assign TTC values to ICS attack techniques defined by MITRE ATT&CK. A similar attempt to assign TTC probability distributions to the MITRE ATT&CK Enterprise domain was made by Xiong et al. [11]. They use a different method than the equation of estimating TTC as described in this paper and focus on the enterprise domain rather than the ICS domain. The method they use is to find resources, such as, academic papers and technical reports that contain information about the time taken for an attack or the probability that it succeeds. They do so with a systematic literature review and evaluate the credibility of the source since they acknowledge that the result is qualitative rather than quantitative. Considering that our method is quantitative, it makes it easier to update the TTC values and allows for more scalability.

The ICS asset that the techniques can be used on is specified in the MITRE ATT&CK framework. In the vulnerability dataset used when estimating the TTC_{ICS}, the vulnerabilities are categorised according to product type. Since the ICS asset and product types do not directly translate, we must find a method of mapping the ICS assets to product type. Otherwise, we are unable to find the correct vulnerabilities in the dataset. A similar method of mapping the MITRE ATT&CK techniques to vulnerabilities is done by the Center for Threat Informed Defense [12]. They use the techniques for describing vulnerabilities in the Common Vulnerabilities and Exposures (CVE) list. Their intention is that this will make it easier for defenders to integrate vulnerabilities into their threat modeling considering that the vulnerabilities themselves are often detailed and technical rather than higher-level. When mapping, they ask the question "what steps are necessary to exploit this vulnerability?" and select a technique based on this.

Background

In the following background sections, we describe the TTC by McQueen et al. [5], the ICS vulnerability dataset used to estimate the TTC [6] and finally the MITRE ATT &CK ICS technique knowledge base [4] on which we apply the TTC estimations.

Time-To-Compromise

In 2006, McQueen et al. published their first paper on the TTC and presented a method of how to estimate it [5]. The application of TTC estimations of ICS used in this paper is built on TTC_{ICS}, which is an extension of the original TTC [3]. The TTC is estimated by combining the time to complete and probability for an attacker to exist in three different processes. In the first process, there is at least one available exploit and one known vulnerability for the component. In process two, there is no known exploit but at least one vulnerability is known. The third process is finding new vulnerabilities and creating new exploits. An attacker is either in process one or two since these are mutually exclusive, but will always continue to be in process three.

The TTC is estimated for four different skill levels of the attacker. These skill level are novice, beginner, intermediate and expert. McQueen et al. make the assumption that a more skilled attacker would have more exploits available to them and therefore the TTC would decrease. The differences between the TTC and TTC_{ICS} are several. First, the dataset of vulnerabilities is different since we use a ICS specific dataset [6]. Next, the method of calculating the number and fraction of exploits available to the hacker based on skill level is updated. The estimated times to complete processes one and two are also updated based on new research. Instead of using a fixed value of Mean-Time-Between-Vulnerabilities (MTBV) in TTC_{ICS} this value is estimated for every type of attack. We have also included the values of CVSS to take into consideration the severity or exploitability of the vulnerability.

Table 1 Product types for vulnerabilities as proposed by Thomas and Chothia for the ICS vulnerability dataset [6]

Product type
AC drive
Access management system
Actuator
CCTV
Charging station
Converter
HMI
Inverter
Network management
Networking
PLC
Power metering
Protection system
Remote I/O
RTU
RTU management
SCADA
Serial server
Smart grid

Table 2 Main categories of vulnerabilities as proposed by Thomas and Chothia for the ICS vulnerability dataset [6]

Category
Default credentials *(Default Logins)*
Denial of service and resource exhaustion *(Denial of Service)*
Exposed sensitive data *(Information Leakage)*
Memory and buffer management *(Memory)*
Permissions and resource access control *(Access Control)*
Privilege escalation and authentication weaknesses *(Authentication)*
Weak and broken cryptography *(Cryptography)*
Web-based weaknesses *(Web)*

Vulnerability Dataset

To make the TTC_{ICS} ICS specific, we use a vulnerability dataset that only includes vulnerabilities for the ICS domain.[1] The dataset includes the product types as seen in Table 1 and the many different categories of vulnerabilities. The eight main categories are shown in Table 2 and these cover 95% of all vulnerabilities [6]. Compared to the more than 185 000 vulnerabilities in the National Vulnerability Database (NVD),[2] the ICS vulnerability dataset includes only 2740 vulnerabilities, after removing rejected ones.

Table 3 Parameters used from the ICS vulnerability dataset [6] for estimating TTC_{ICS}

Parameter	Description
`cvss_exploitability_score`	CVSS exploitability score
`cvss_base_score`	CVSS base score
`u_new_cat`	Category of the CVE as detected by the dataset creators
`u_other_cat`	Details of "Other" category of the CVE
`u_product_type`	Type of product
`u_sys_created`	ICS advisory creation date

Besides categorizing the vulnerabilities according to product type and category of vulnerability, the dataset also includes other important data for estimate the TTC, as seen in Table 3. Considering that CVSS version 2 type of vulnerabilities does not include a CVSS exploitability score, we also use the value of the CVSS base score when estimating the TTC. The ICS advisory creation date is used when estimating the Mean-Time-Between-Vulnerabilities (MTBV).

Techniques Knowledge Base

MITRE ATT&CK [4] is a knowledge base of tactics and techniques for attacking systems. One of the intentions of the knowledge base is to function as a framework when assessing the security of a system. When developing threat models it is useful to be able to refer to established tactics and techniques since it makes it easier to share information. With the work presented in this paper we aim to contribute to MITRE ATT&CK by adding the estimate of TTC_{ICS}. MITRE ATT&CK includes tactics and techniques for the enterprise, mobile and ICS domains. In this paper we only focus on the tactics and techniques of the ICS domain. The MITRE ATT&CK ICS matrix shows the relation between the 78 different techniques categorized by 12 different tactics.

The tactics of MITRE ATT&CK answers the question *why* an attacker wants to perform an action. The techniques answers the question of *how* they perform the action. For example, an attacker may use the technique *Exploit Public-Facing Application* as a *how* to perform the tactics of *Initial Access*. The *Initial Access* is *why* they wanted to perform the technique. There are some techniques that can be used for several tactics. Besides the tactics and techniques, the knowledge base also includes which asset the technique applies to. MITRE recognizes seven assets in ICS networks: Control Server, Data Historian, Engineering Workstation, Field Controller/RTU/PLC/IED, Human–Machine Interface,

[1] ICS Vulnerability Dataset, https://github.com/UoB-RITICS/esorics2020-dataset [Accessed 13 Feb 2023].

[2] National Vulnerability Database, https://nvd.nist.gov/ [Accessed 13 Feb 2023].

Input/Output Server and Safety Instrumented System/Protection Relay.

Method

The method of applying the TTC_{ICS} estimate to MITRE ATT&CK knowledge base is to first map the techniques of MITRE ATT&CK to the product types and vulnerability categories of the ICS vulnerability dataset. By mapping the techniques in this way we can calculate the values of the parameters needed for the TTC_{ICS} estimation. Thereafter, we estimate the TTC_{ICS} values for the techniques.

Mapping of Techniques

The method of estimating TTC_{ICS} for an attack requires that the product type and category of vulnerability are known. This is because we use a dataset of vulnerabilities for ICS that categories the vulnerabilities according to product type and category of vulnerability. When estimating the TTC_{ICS}, we want to be able to find the specific vulnerabilities applicable for that product type and vulnerability. We assign the product type and vulnerability category for each technique as a way of mapping them. First we translate the ICS asset, which MITRE specifies for each technique, to product type and then we read the description of the technique to assign the most appropriate category of vulnerability that the technique is used for.

The techniques in the MITRE ATT&CK knowledge base specify which ICS asset that the technique relates to. For 13 of the techniques, MITRE does not specify an ICS asset and then we do not map it or estimate the TTC_{ICS}. Two of the seven ICS assets defined by MITRE ATT&CK are not present as a product type in the ICS vulnerability dataset. This is because the assets, Engineering workstation and Data historian, are not necessarily ICS specific. To still be able to align the two ICS assets to the ICS vulnerability dataset, we search through the description of the vulnerability for the keywords "workstation" and "historian". We only search the description of those vulnerabilities in the dataset that do not already have an assigned product type. The product type for these vulnerabilities are set to "NULL".

When mapping the techniques to vulnerability categories, we ask the questions "Which of the vulnerability categories present in the dataset is the most aligned to the described technique?" In addition we also look at the mitigations for the technique as suggested by MITRE to help us map correctly. For example, if the mitigation for the attack is "Access Management" the appropriate vulnerability category may be "Access Control". We first consider the main eight categories as seen in Table 2. If these categories are not valid vulnerabilities for the technique, we try to align it to any of the "Other" categories. The "Other" are 21 smaller categories that have been manually selected by the creators of the ICS vulnerability dataset.

Estimation of TTC_{ICS}

When using the estimate for TTC_{ICS}, some of the variables are fixed and some needs to be adjusted for the specific technique that we want to estimate the TTC_{ICS} for.

In process 1, as seen in Eq. 1 below, the value of k is fixed to 2740 since that is the total number of entries in the ICS vulnerabilities dataset [6]. This value will change if the number of vulnerabilities of the dataset changes. The aim is to have the total number of vulnerabilities for all ICS assets.

The values of v, $c2$ and $c3$ depends on the number of vulnerabilities that exist for the specific technique. For m, the value is derived by searching for exploits in the Metasploit database. The Metasploit database ranks their exploits depending on how reliable they are, and we use this ranking to assume which exploit would be usable by an attacker with different skill levels. For example, if an exploit is of ranking "Excellent", we assume that all four skill levels of an attacker can use the exploit. However, if the exploit has a "Low" ranking, we assume that only expert attacker would be able to modify and use the exploit successfully. To find the number of exploits with access control as part of the description, at skill level excellent, the following search command is used:

```
search description:"access control"
type:exploit rank:excellent
```

$$P_1 = 1 - e^{-vm/k}, t_1 = 1 * ((10/c2 + 3,9/c3)2)days \quad (1)$$

where v is the total number of vulnerabilities of that type component for that type attack [6], m is the number of exploits available to the hacker based on skill level, k is the total number of vulnerabilities [6], $c2$ is the average CVSS of version 2 vulnerabilities and $c3$ is the average CVSS of version 3 vulnerabilities.

The probability of an attacker to be in process 2 is the inverse of the probably of being in process 1, as seen in Eq. 2. The time taken to complete process 2 is estimated to be 37 days for a novice, 27 days for a beginner, 16 days for an intermediate and 6 days by an expert attacker. These values are found from a report which estimates that the time taken to develop an exploit for a known vulnerability is usually between 6 to 37 days with a median value of 22 days [13]. We derive our values by dividing the range of days

Table 4 The number and fraction of vulnerabilities that are exploitable based on skill level [3]

Skill level	CVSS exploitability range	Exploitable vulnerabilities	Fraction of exploitable mulnerabilities
Expert	0.1–3.9	1916	1
Intermediate	0.1-3	966	0.50
Beginner	0.1–2.1	455	0.24
Novice	0.1–1.2	105	0.05

between the four different skill levels of an attacker. Even though the report is not ICS specific, we assume that the time that it takes to develop an exploit is the same for the ICS domain as for any other domain.

$$P_2 = e^{-vm/k} = 1 - P_1 \tag{2}$$

Considering that process 3 is running in parallel to process 1 and 2, since an attacker could continuously try to find new vulnerabilities and build exploits, we do not estimate the probability of an attacker to be in process 3. For the time taken to complete process 3, we firstly consider the time that it would take an attacker to create an exploit for a known vulnerability and the time t_2. Secondly we use the fraction of vulnerabilities that are exploitable (f), as seen in Table 4, and the Mean Time Between Vulnerabilities (b) to estimate the time it would take to find a new vulnerability and create an exploit for it.

The fraction of vulnerabilities that are exploitable (f) is estimated based on the CVSS exploitability score considering all the vulnerabilities in the vulnerability dataset. This gives us an estimate of which fraction of the vulnerabilities is exploitable for each skill level. 1916 of the records had the score assigned from values 0.3 to 3.9, but we consider the theoretical maximum range of 0.1 to 3.9 as seen in Table 4. Each of the vulnerabilities have a creation date of when the vulnerability became a CVE entry. We use the average time between each creation date to estimate the Mean Time Between Vulnerabilities (b). In reality the value shows how often a vulnerability is reported for a specific product type and category of vulnerability, but we use it as an indication of how often vulnerabilities are found.

$$t_3 = (f' - 0.5) * b + t_2 \tag{3}$$

where t_2 is the number of days taken to develop a new exploit, f' is the inverse of the fraction of vulnerabilities that are exploitable based on skill level and b is the Mean-Time-Between-Vulnerabilities (MTBV) in days as calculated from the ICS advisory creation day.

$$T = t_1 * P_1 + t_2 * (1 - P_1) * (1 - u) + t_3 * u * (1 - P_1) \tag{4}$$

Table 5 Mapping of the MITRE ATT &CK ICS Assets to the Product Types of the ICS Dataset

ATT &CK ICS asset	Product type
HMI	HMI
Safety instrumented system/protection relay	Protection system
Control server	RTU management
Field controller/RTU/PLC/IED	RTU and PLC
Input/output server	Serial server
Engineering workstation	Description includes "workstation"
Data historian	Description includes "historian"

where T is the expected TTC, $u = (1\text{-f})^v$, which is the probability that Process 2 is unsuccessful where v is the number of vulnerabilities and f is fraction of vulnerabilities that are exploitable for a specific skill level. Also, u=1 if v=0.

Results

The mapping between the MITRE ATT &CK ICS assets and the product types of the ICS dataset are shown in Table 5.

The reasoning when mapping the techniques to vulnerability categories is presented in Appendix 7. There are 78 techniques defined by MITRE ATT&CK, but as described in Sect. "Mapping of Techniques", 13 have been removed since they did not define an ICS asset affected by the technique. We are thus left with 65 techniques. We can see some patterns, such as that Initial Access techniques mostly relate to access control vulnerabilities and execution and persistence mostly relate to command or code injections. We also see how most Collection techniques relate to information leakage vulnerabilities and Lateral Movement is caused by access control issues. In Table 6 we show an overview of the different vulnerability categories with ICS assets that were found when mapping the MITRE ATT&CK techniques.

Considering that the calculation of TTC_{ICS} follows the same method, we do not show the result of all estimated values. Instead we focus on the most common vulnerability categories that we found when mapping the techniques. These are *Access Control*, *Information Leakage* and *Denial of Service*.

We also estimate the TTC_{ICS} specifically for the ICS assets *HMI* and *protection system*. This is because the HMI is most commonly place on the edge of the network and sometimes accessible remotely whereas the protection system is often placed further within the network. By looking at two different types of assets while considering three different vulnerability categories we will be able to showcase the estimation of TTC_{ICS}. The results on the estimated TTC_{ICS} values are found in Table 7, rounded to the nearest day.

Table 6 The resulting vulnerability categories found per ICS asset type after mapping the MITRE ATT&CK Techniques to the ICS vulnerability dataset

	HMI	Protection System	RTU Management	RTU and PLC	Serial Server	"workstation"	"historian"
Access Control	●	●	●	●	●	●	●
Authentication	●	●		●			
Code Injection	●	●		●	●	●	
Command Injection	●	●	●	●	●		●
Cryptography			●				
Default Logins	●	●	●	●	●	●	●
Denial of Service	●	●	●	●	●	●	●
Direct Shell Command	●		●			●	●
DLL Hijacking						●	
Hidden Functionality	●	●	●	●	●		●
Information Leakage	●	●	●	●		●	●
Memory				●			
Web	●						

Table 7 The estimated TTC_{ICS} values per skill level (in days) for the HMI and protection system assets considering vulnerability categories Access Control, Information Leakage and Denial of Service

Product type	Vulnerability category	TTC_{ICS} novice	TTC_{ICS} beginner	TTC_{ICS} Intermediate	TTC_{ICS} expert
HMI	Access control	3594	98	15	6
HMI	Information leakage	2501	59	16	6
HMI	Denial of service	716	27	15	6
Protection system	Access control	5739	466	48	6
Protection system	Information leakage	37	27	16	6
Protection system	Denial of service	5823	472	50	6

The table shows the estimated TTC_{ICS} for different combinations of the product types HMI and Protection systems with the category of vulnerabilities Access Control, Information Leakage and Denial of Service. These combinations represent MITRE ATT&CK techniques as defined in the next section. For each combination, the TTC_{ICS} has been estimated per skill level of the attacker. The Excel document where the estimations has been performed and instructions for how to create them can be found on GitHub[3].

The techniques mapped to *Access Control* and HMI are Manipulation of View, Internet Accessible Device, Exploitation of Remote Services, Remote Services and Lateral Tool Transfer. The techniques mapped to *Information Leakage* and HMI are Data from Information Repositories, Man in the Middle, Monitor Process State, Point & Tag Identification, Screen Capture, Network Connection Enumeration, Remote System Discovery. For *Denial of Service* there are only three: Loss of View, Modify Alarm Settings and Service Stop.

The techniques mapped to *Access Control* and protection systems are Program Upload, Change Operating Mode, Internet Accessible Device and Program Download. For *Information Leakage* and protection systems there are techniques Automated Collection, Monitor Process State, Remote System Discovery and Remote System Information Discovery. Finally for *Denial of Service* and protection systems there are Alarm Suppression, Denial of Service and Modify Alarm Settings.

We can see from Table 7 that generally the techniques with *Access Control* for HMI have a higher TTC_{ICS} compared to *Information Leakage* and *Denial of Service*. It is expected that it would take more time to gain access on an HMI compared to cause an DoS or even gain information from, for example, sniffing. The type of vulnerability with lower TTC_{ICS} is *Denial of Service*, which is one of the common types of attacks against ICS [14]. The data also suggest that the TTC_{ICS} is higher for protection systems. This supports the idea that protection systems and other OT-systems, as compared to more standard IT-systems, are more difficult to access and understand for attackers than the contradicting hypothesis that OT systems are generally built with poorer security. We must however also acknowledge the data set that we have with 148 vulnerabilities for HMI as compared to 32 for protection systems.

It appears that for the protection system, the TTC_{ICS} is similar for *Access Control* and *Denial of Service*. A possible explanation to this could be that protection systems are sometimes completely lacking access control. Poor access control was found to be a common weakness in new ICS software in 2009–2010 [15].

[3] TTC_{ICS}, https://github.com/EngLi/ttc-ics, [Accessed 13 Feb 2023].

Discussion

From the results we can see that the TTC_{ICS} for protection systems information leakage stands out as low. The reason for this is that the vulnerabilities in the dataset were reported on the same day and therefore the MTBV is 0. For the estimation, this indicates that new vulnerabilities are very quickly being discovered, even though that is not the case since the vulnerabilities were only 4. Improvements on the tool would be to in these cases use another MTBV, such as the average MTBV for those types of vulnerabilities for any given asset. In this case the average MTBV for all vulnerabilities of type *Information Leakage* is 14, rounded to the closest day. This is still a lot lower MTBV compared to the other estimated TTCs, but using the MTBV of 14 changes the TTC_{ICS} from 37, 27, 16 and 6 days to 258, 44, 17 and 6 days. It is also noticeable from the results that regardless of the type of asset or category of vulnerability, an expert is able to compromise it in 6 days.

The result of the final TTC estimate is greatly dependent on how we map the techniques of attack to vulnerability category type. It is also dependent on how we map the ICS assets to the assets present in the ICS vulnerability dataset. Many of the techniques maps to the same vulnerability category and product type. This means that we will see the same TTC_{ICS} for many different techniques. This is due to the lack of granularity and we could achieve a more specific value of TTC_{ICS} per technique if the techniques were more specific. We believe, however, that it is a good start to get a TTC_{ICS} value per technique even though many of them will be the same. Two of the ICS assets from MITRE ATT&CK were not present in the vulnerability dataset and in this case we instead searched the description of the vulnerability. An improvement would be to continue the categorisation work of the dataset and assign asset to each vulnerability. Many of the assets in the vulnerability dataset did not have an asset defined.

We acknowledge that the ICS vulnerability dataset must be accurate to result in a correctly estimated TTC value, which places the trust in already conducted research. We also trust that the method of estimating the TTC_{ICS} is accurate. Considering that TTC_{ICS} combines data and frameworks in a novel way, we are unable to compare it directly to other existing TTC approaches. For example, the TTC by McQueen etl. al. [5] does not look at specific attacks, but estimates the overall TTC for components. Instead, the results of the estimate has been evaluated by comparing it to other research results. However, future work includes further validation by interviewing experts in the ICS domain. The experts would be able to compare their assessment of the TTC with the results in this article. Other future work is to create an automated tool to estimate the TTC instead of using an Excel sheet to do the calculations. The tool could automatically be updated if the MITRE ATT&CK techniques or the vulnerability dataset would change to stay up-to-date.

In a survey where ethical hackers were asked about how long it takes for them to perform certain tasks, it was found that it takes the majority are able to collect data after gaining access to a system in between 1 and 5 h [16]. The same survey shows that it takes the majority 1–5 h to break into an environment after finding an exposure. For the estimated TTC_{ICS}, we found that it would take an expert 6 days to compromise the asset. The discrepancy in TTC could be because the survey of ethical hackers were not ICS specific and this could indicate that typical IT systems are faster to compromise. The reason could also be related to that the time scale is not calibrated between the two data sets.

Appendix A: Mapping MITRE ATT &CK ICS Techniques to Vulnerability Type

Initial Access

- **Exploit Public-Facing Application** The public-facing application can be accessible remotely via the Internet when using a web browser and therefore we consider the category of vulnerability as *Web*.
- **Exploitation of Remote Services** This vulnerability does not have to be external facing, the attacker can, for example, move laterally from the IT network. The technique gives both initial access to the ICS environment and lateral movement within so we consider the category as *Access Control*.
- **External Remote Services** This technique exploits remote services that run externally. Considering that the external remote services means an initial access to a service that is should not be accessible by an attacker, we consider this to be an *Access Control* vulnerability.
- **Internet Accessible Device** Even though the main weakness is that the device is accessible via the Internet, the vulnerability itself is that the access control is not implemented correctly in case of an attach. Therefore we consider it to be a *Access Control* category.
- **Remote Services** By exploiting remote services, the attacker is typically using vulnerabilities of the type *Access Control*, since they have been able to bypass this.
- **Replication Through Removable Media** For this technique, the attacker would place malware on removable media, for example a USB, and physically connect it to the system. Since this technique could place any number of different types of vulnerabilities on the removable media, we are unable to categorize it.

- **Rogue Master** A rouge master can be used for sending malicious control signals or to disrupt traffic from or to the actual master. There are many different vulnerabilities that could lead to this type of attack and therefore we are unable to categorize it.
- **Spearphishing Attachment** Spearphishing is a targeted attack where malware is send to the victim. Because the malware could target any number of different vulnerabilities, we cannot categorize the technique.
- **Supply Chain Compromise** For supply chain attacks, a malicious actor has compromised the component before it reaches the user. We consider this to be of vulnerability category *Hidden Functionality*.
- **Transient Cyber Asset** These assets are used temporarily to, for example, transfer data or troubleshoot issues by directly connecting them to an asset in the system. An attacker could gain malicious access via the asset and therefore we consider this as an *Access Control* vulnerability.
- **Wireless Compromise** Using the wireless network to gain access to the network we consider to be of vulnerability type *Access Control*.

Execution

- **Change Operating Mode** By changing the operating mode on a controller, the attacker can gain more execution control. We consider this an *Access Control* vulnerability.
- **Command-Line Interface** We categorize using the command-line interface to perform attacks to be of vulnerability *Direct Shell Command*.
- **Execution through API** Malicious use of Application Program Interfaces (APIs) we categorize as a *Command Injection* vulnerability.
- **Hooking** By hooking onto API processes, the attacker can redirect calls. MITRE mentions that Windows APIs are usually stored in dynamic-link libraries (DLLs). We therefore categorize it as a *DLL Hijacking* vulnerability.
- **Modify Controller Tasking** An attacker can modify the tasks of a controller to change the execution flow to run their own programs. We categorize this as a *Code Injection* vulnerability.
- **Native API** A native API allows for interaction with the OS. When used maliciously categorize as a *Command Injection* vulnerability.
- **Scripting** We categorize using a script maliciously to be of vulnerability category *Code Injection*.
- **User Execution** The attacker can trick the user to execute malicious code. We consider this to be of vulnerability category *Code Injection*.

Persistence

- **Modify Program** We consider that modifying a program maliciously to be categorized as a *Code Injection* vulnerability.
- **Module Firmware** Inserting malicious firmware on an asset we consider as a *Code Injection* vulnerability.
- **Project File Infection** Infecting project files we consider to be categorized as a **Code Injection** vulnerability.
- **System Firmware** An attacker can utilize the system firmware update and update it with its own malicious firmware. We consider this to be a **Code Injection** vulnerability.
- **Valid Accounts** By utilizing a valid account, often by using the default credentials, the attacker can gain increased privileges. We consider this to be of vulnerability type *Default Logins*.

Privilege Escalation

- **Exploitation for Privilege Escalation** We consider that being able to escalate privileges is of vulnerability category *Authentication*.
- **Hooking** *See Section* "Execution".

Evasion

- **Change Operating Mode** *See Section*"Execution".
- **Exploitation for Evasion** The attacker can use any number of vulnerabilities for exploiting therefore we are unable to categorize this technique.
- **Indicator Removal on Host** Removing or deleting traces of yourself as an attacker is not a type of vulnerability therefore we cannot categorize this technique.
- **Masquerading** The act of hiding files or changing names of them to evade detection we consider to be of category *Control of Filename/Path*.
- **Rootkit** Since the rootkit is used to hide malicious behavior by intercepting API calls we consider this as a *Command Injection* type of vulnerability.
- **Spoof Reporting Message** Spoofing can be avoided with proper cryptographic security measures, therefore we consider this to be of vulnerability type *Cryptography*.

Discovery

- **Network Connection Enumeration** Finding any sensitive information that can be used by a malicious attacker we consider to be of vulnerability category *Information Leakage*.

- **Network Sniffing** Finding any sensitive information that can be used by a malicious attacker we consider to be of vulnerability category *Information Leakage*.
- **Remote System Discovery** Finding any sensitive information that can be used by a malicious attacker we consider to be of vulnerability category *Information Leakage*.
- **Remote System Information Discovery** Finding any sensitive information that can be used by a malicious attacker we consider to be of vulnerability category *Information Leakage*.

Lateral Movement

- **Default Credentials** We categorize default credentials as vulnerability category *Default Logins*.
- **Exploitation of Remote Services** *See Section*"Initial Access".
- **Lateral Tool Transfer** Moving tools maliciously throughout the system is not caused by a specific vulnerability. But, since moving them would require access, we consider this to be an *Access Control* vulnerability category.
- **Program Download** Downloading programs maliciously is not caused by a specific vulnerability. But we consider that the attacker would need access and we therefore categorize this technique as a *Access Control* category of vulnerability.
- **Remote Services** *See Section* "Initial Access".
- **Valid Accounts** *See Section* "Persistence"

Collection

- **Automated Collection** The collection of information regarding the system we consider to be of vulnerability type *Information Leakage*.
- **Data from Information Repositories** We consider that retrieving data from information repositories is of vulnerability type *Information Leakage*.
- **Detect Operating Mode** Any type of information that can help the attacker, such as the operating mode, we consider to be of category *Information Leakage*.
- **I/O Image** Maliciously reading the input and output of a PLC we consider to be of category *Information Leakage*.
- **Man in the Middle** A Man in the Middle attack we consider to be of category *Information Leakage* since the attacker is sniffing data in the middle of two communication points.
- **Monitor Process State** An attack to find information about the process state we consider to be of category *Information Leakage*.
- **Point & Tag Identification** Collecting points, such as, memory locations and tags, which are identifiers, we consider to be of vulnerability category *Information Leakage*.
- **Program Upload** Uploading a program to be able to gain information we consider to be of vulnerability *Access Control* since the attacker would require access to upload it.
- **Screen Capture** Gaining information maliciously by capturing a screen we consider to be of vulnerability type *Information Leakage*.

Command and Control

- **Commonly Used Port** Using common ports for communication to hide the traffic flow is not exploiting a vulnerability therefore we cannot categorize it.
- **Standard Application Layer Protocol** Using application layer protocols for malicious communication is not caused by exploiting a vulnerability and therefore we cannot categorize it.

Inhibit Response Function

- **Activate Firmware Update Mode** Activating the firmware update mode is not an attack cause by a vulnerability, therefore we are unable to categorize it.
- **Alarm Suppression** Suppressing the alarms and therefore causing this service to not work properly we categorize as a *Denial of Service* type of vulnerability.
- **Block Command Message** Blocking command messages we consider as a *Denial of Service* type of vulnerability.
- **Block Reporting Message** Blocking reporting messages we consider as a *Denial of Service* type of vulnerability.
- **Block Serial COM** Blocking Serial COM we consider as a *Denial of Service* type of vulnerability.
- **Data Destruction** Destroying data is not caused by a type of vulnerability and we can therefore not categorize it.
- **Denial of Service** We categorize Denial of Service as a *Denial of Service* type of vulnerability.
- **Device Restart/Shutdown** Restarting or shutting down a devices renders it unavailable and we therefore categorize it as a *Denial of Service* type of vulnerability.
- **Manipulate I/O Image** Manipulation of the I/O image can be performed by for example memory manipulation, therefore we categorize the vulnerability as *Memory*.
- **Modify Alarm Settings** When alarm settings are modified so that the intended alarm function is disabled we categorize it as a *Denial of Service* type of vulnerability. When an operator is unable to see the alarms, they are unable to respond to the hazardous event, which can cause damage to the system.
- **Rootkit** *See Section* "Evasion".

- **Service Stop** We categorize stopping a service as a *Denial of Service* type of vulnerability since it will cause the service to not be reached.
- **System Firmware** *See Section* "Persistence".

Impair Process Control

- **Brute Force I/O** Bruteforcing the I/O can cause instability and possibly failure of a process. We therefore categorize it as a *Denial of Service* type of vulnerability.
- **Modify Parameter** We consider modifying parameters maliciously as a *Command Injection* type of vulnerability.
- **Module Firmware** *See Section*"Persistence".
- **Spoof Reporting Message** *See Section*"Evasion".
- **Unauthorized Command Message** Sending unauthorized command messages can be used as a technique and we categorize it as an *Authentication* type of vulnerability.

Impact

- **Loss of View** Causing a loss of view of ICS equipment we consider to be of a *Denial of Service* type of vulnerability.
- **Manipulation of View** If an attacker can manipulate the view we assume that they have access and therefore we consider it to be a *Access Control* type of vulnerability.

Acknowledgements This project has been funded by the Swedish national research Center on Resilient Information and Control Systems (RICS).

Funding Open access funding provided by Royal Institute of Technology. This project has been funded by the Swedish national research Center on Resilient Information and Control Systems (RICS).

Data availability The document where the estimations has been performed and instructions for how to create them can be found on Github: https://github.com/EngLi/ttc-ics.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Micro T. New Critical Infrastructure Facility Hit by Group Behind TRITON (2019). https://www.trendmicro.com/vinfo/pl/security/news/cyber-attacks/new-critical-infrastructure-facility-hit-by-group-behind-triton Accessed 13 Feb 2023
2. Andreeva O, Gordeychik S, Gritsai G, Kochetova O, Potseluevskaya E, Sidorova SI, Timorin AA. Industrial Control Systems Vulnerability Statistics (2017). https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2016/07/07190426/KL_REPORT_ICS_Statistic_vulnerabilities.pdf Accessed 13 Feb 2023
3. Rencelj Ling E, Ekstedt M. Estimating the Time-To-Compromise of Exploiting Industrial Control System Vulnerabilities. In: Proceedings of the 8th International Conference on Information Systems Security and Privacy - ICISSP, pp. 96–107. SciTePress, Portugal (2022). https://doi.org/10.5220/0010817400003120.INSTICC
4. MITRE ATT &CK: ICS Techniques (2022). https://attack.mitre.org/techniques/ics/ Accessed 13 Feb 2023
5. McQueen MA, Boyer WF, Flynn MA, Beitel GA. Time-to-compromise model for cyber risk reduction estimation. In: Gollmann D, Massacci F, Yautsiukhin A, editors. Quality of Protection. Boston, MA: Springer; 2006. p. 49–64.
6. Thomas RJ, Chothia T. Learning from vulnerabilities - categorising, understanding and detecting weaknesses in industrial control systems. In: Computer Security. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64330-0_7
7. Nzoukou W, Wang L, Jajodia S, Singhal A. A unified framework for measuring a network's mean time-to-compromise. In: 2013 IEEE 32nd International Symposium on Reliable Distributed Systems, pp. 215–224 (2013). https://doi.org/10.1109/SRDS.2013.30
8. Zieger A, Freiling F, Kossakowski K. The β-time-to-compromise metric for practical cyber security risk estimation. In: 2018 11th International Conference on IT Security Incident Management IT Forensics (IMF), pp. 115–133 (2018). https://doi.org/10.1109/IMF.2018.00017
9. FIRST: Common Vulnerability Scoring System SIG (2022). https://www.first.org/cvss/ Accessed 13 Feb 2023
10. Leversage DJ, Byres EJ. Estimating a system's mean time-to-compromise. IEEE Secur Privacy. 2008;6(1):52–60. https://doi.org/10.1109/MSP.2008.9.
11. Xiong W, Hacks S, Robert L. A method for assigning probability distributions in attack simulation languages. Complex Syst Inform Model Q. 2021;2021:55–77. https://doi.org/10.7250/csimq.2021-26.04.
12. Center for Threat Informed Defense: Using MITRE ATT &CK to Describe Vulnerabilities. https://github.com/center-for-threat-informed-defense/attack_to_cve Accessed 13 Feb 2023
13. Ablon L, Bogart A. Zero Days, Thousands of Nights: The Life and Times of Zero-Day Vulnerabilities and Their Exploits. RAND Corporation, Santa Monica, CA (2017). https://doi.org/10.7249/RR1751
14. Stouffer K, Pillitteri V, Lightman S, Abrams M, Hahn A. Guide to Industrial Control Systems (ICS) Security (2015). https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf Accessed 13 Feb 2023
15. Homeland Security: Common Cybersecurity Vulnerabilities in Industrial Control Systems (2011). https://www.cisa.gov/uscert/sites/default/files/recommended_practices/DHS_Common_Cybersecurity_Vulnerabilities_ICS_2010.pdf Accessed 13 Feb 2023

16. Bromiley M. Think Like a Hacker: Inside the Minds and Methods of Modern Adversaries (2022). https://s3.us-east-2.amazonaws.com/s3.bishopfox.com/prod-1437/Documents/Reports/SANS-Report-Hacker-Survey-2022.pdf Accessed 13 Feb 2023