Automated ICS template for STRIDE Microsoft Threat Modeling Tool

Mike Da Silva
CEA-Leti, Université Grenoble Alpes
Grenoble, France
mike.dasilva@cea.fr

Maxime Puys
CEA-Leti, Université Grenoble Alpes
Grenoble, France
maxime.puys@cea.fr

Pierre-Henri Thevenon
CEA-Leti, Université Grenoble Alpes
Grenoble, France
pierre-henri.thevenon@cea.fr

Stéphane Mocanu
Laboratoire d'Informatique de Grenoble
Univ. Grenoble Alpes, CNRS, Inria, Grenoble-INP
Grenoble, France
stephane.mocanu@inria.fr

Nelson Nkawa
Laboratoire d'Informatique de Grenoble
Univ. Grenoble Alpes, CNRS, Inria, Grenoble-INP
Grenoble, France
nelson.nkawa@inria.fr

ABSTRACT

Industrial Control Systems (ICS) are specific systems that combine information technology (IT) and operational technology (OT). Due to their interconnection and remote accessibility, they become a target for cyberattacks. As a result of their complexity and heterogeneity in terms of devices and communication protocols, specific security controls and risk analysis methods need to be developed. In particular, in order to reduce the effort of deployment of risk analysis on such complex systems, automated methods need to be provided. This paper deals with automation of the risk identification process for ICS using the STRIDE threat modeling framework. We extend the well-known STRIDE modeling tool, namely Microsoft Threat Modeling Tool (MTMT), with an incremental template dedicated to ICS and provide additional tools to automate the analysis using specific vulnerability extraction from Internet CVE databases.

KEYWORDS

MTMT, Security, Cybersecurity, Risk assessment, Risk analysis, SCADA, ICS, IT, OT

ACM Reference Format:
Mike Da Silva, Maxime Puys, Pierre-Henri Thevenon, Stéphane Mocanu, and Nelson Nkawa. 2023. Automated ICS template for STRIDE Microsoft Threat Modeling Tool. In *The 18th International Conference on Availability, Reliability and Security (ARES 2023), August 29–September 01, 2023, Benevento, Italy*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3600160.3605068

1 INTRODUCTION

Industrial Control Systems (ICS) are used to monitor and control physical processes such as energy production and distribution, manufacturing or transport systems. They are often loosely referred as Supervisory Control And Data Acquisition (SCADA) systems. For decades, they have been deployed on dedicated and isolated proprietary networks and they were considered protected from remote threats thanks to their isolation. However, due to their recent interconnection on the Internet as part of Industry 4.0 they are s caused by various opponents such as criminal groups or foreign states. Stuxnet malware in 2011 [14] was the first important attack revealed to the media. As of now, cybersecurity incidents involving ICS are regularly discovered and the number of attacks against industrial facilities is continuously growing. As the frequency of such attacks is increasing, securing ICS became a priority for governmental agencies and, consequently, cybersecurity controls are now mandatory for some critical industrial activities. The first step of securing and certifying industrial devices or systems is the cybersecurity risk assessment.

Several risk assessment methods already exist and are used to evaluate the system's security as EBIOS Risk Manager [4], STRIDE [18], DREAD [10], LINDDUN [9], PASTA [19]. However, such risk analysis methods are generic and applying them on specific types of systems can quickly become a tedious process. Therefore, in some application domains, developing dedicated templates and databases will help the modeling of systems during analysis. For instance, there are templates for medical devices or services deployed in Azure cloud. Specifically, industrial systems are mainly composed of proprietary hardware (for liability reasons), and, de facto, there is a large choice of models that accomplish the same function (e.g., PLCs or motor drivers). Therefore, it is difficult to maintain an up-to-date database of vulnerabilities which are specific to each particular model. In the same way, industrial devices usually communicate with multiple industrial network flow types (both secured and unsecured) at the same time.

Contributions: In this paper, we propose a toolkit to simplify STRIDE modeling for industrial systems with an automated generation of threats based on known vulnerabilities. This toolkit includes an extendable database of ICS components permitting both to aggregating knowledge on ICS devices and to model them into Microsoft Threat Modeling Tool [5] (MTMT). MTMT provides a user-friendly graphical environment to perform STRIDE-per-interaction analysis. According to Adam Shostack [18], STRIDE-per-interaction is an approach to threat enumeration that considers tuples of (*origin*, *destination*, *interaction*) and enumerates threats against them. In summary, our contributions are the following:

- An extendable database of ICS components to model specific industrial architectures and threats;
- An automated tool to generate MTMT templates from the database;
- An improved methodology allowing to gather CVE (Common Vulnerability Exposures) from system's devices then matching their impact to STRIDE threats in MTMT.

Outline: The rest of the paper is organized as follows. Section 2 describes background knowledge and state-of-the-art. Section 3 presents an industrial use case to showcase our contributions. Then, we introduce our incremental database for ICS components and a tool to generate MTMT template from the database in Section 4 and a MTMT CVE modeling in Section 5. Finally, Section 6 concludes the paper with a discussion of future works.

concludes the paper with some discussion of where the synchronous languages will be in the future

2 BACKGROUND AND STATE-OF-THE-ART

In this section, we recall some background information on the STRIDE risk analysis methodology alongside Microsoft Threat Modeling Tool and detail the state-of-the-art in relation to our contributions.

2.1 STRIDE

STRIDE is a cybersecurity risk analysis methodology developed in 2009 by Loren Kohnfelder and Praerit Garg [13], and adopted by Microsoft in their Security Development Lifecycle (SDL). Supported by Microsoft, this methodology is famous among manufacturers and solution providers. STRIDE stands for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege. Conversely to other methodologies such as EBIOS focusing on assets to protect, STRIDE is based on threats. These threats, and their security objectives in brackets, are defined by the method [18] as:

- **Spoofing (Authentication):** Pretending to be something or someone other than yourself.
- **Tampering (Integrity):** Modifying something on disk, on a network, or in memory.
- **Repudiation (Non-repudiation):** Claiming that you didn't do something or were not responsible. Repudiation can be honest (truthful claim) or false (a lie).
- **Information disclosure (Confidentiality):** Providing information to someone not authorized to see it.
- **Denial of service (Availability):** Absorbing resources needed to provide service.
- **Elevation of privilege (Authorization):** Allowing someone to do something they are not authorized to do.

2.2 Microsoft Threat Modeling Tool

Microsoft proposes a graphical tool allowing to automate the STRIDE methodology: Microsoft Threat Modeling Tool (MTMT). In this tool, each system to analyze is called a model and is depicted as a flow diagram, with components linked together by directional arrows representing communications (at any relevant level). Designing a model requires a template, that is, a library of components/devices/protocols for which threats have been already identified individually (for example, a protocol which will be vulnerable in integrity or an IoT device known for its lack of authentication). Such components are called stencils and can be derived from each other. Then, a functionality allows to generate automatically, from the template and the diagram data, the list of threats contained by the system according to all threat types supported by STRIDE (spoofing, tampering, etc.).

2.3 Related Work

Our work brings together two areas usually treated separately. So, we divided this section in two parts. First, we present work which maps CVE and STRIDE and then work on applying STRIDE for ICS.

2.3.1 *Mapping Common Vulnerability Exposures (CVE) and STRIDE.*

To the best of our knowledge, there is no complete process for linking CVE to STRIDE. In 2021, Honkaranta et al. [8] proposed a method for matching CWE (Common Weakness Enumeration) and STRIDE. In this article, the authors show three different techniques to map CWE with STRIDE. The technique which appears the most efficient relies on CWE's technical impact and scope. In parallel, several AI-based techniques exist in the literature [6, 11, 20, 21] to map CVE with CWE; all based more or less on text processing.

2.3.2 *STRIDE for ICS.*

In 2017, Khan et al. [12] have presented a method in five steps to a systematic application of STRIDE for Cyber-Physical Systems (CPS). Their method is applied through a synchronous islanding use case that allows, in the same time, to explain the method and show its relevance. The first step consists in decomposing the system into components. Authors choose to not consider physical components because they assume that such components are not susceptible to cyberattacks. Then, system components are plotted in a data-flow diagram. In a third step, authors proposed to identify attacker intentions, named threat consequences (e.g., circuit breaker closure in non-synchronized state), and after a classical STRIDE-per-element approach is realized matching threats discovered with threat consequences. Finally, vulnerabilities are identified from threats (e.g., lack of authentication) and a remediation strategy is planned. In a similar way, Asif et al. [2] apply STRIDE approach in an IoT system with a use case based on precision agriculture.

In 2021, FLa et al. [7] have proposed their template for MTMT. They provide a smart grid dedicated template which is, as mentioned before, mainly composed by stencils and threats. To build their smart grid template, they created stencils which are representative, according to an expert's opinion, of a generic smart grid. Threats emerge from a previous work and are derived from literature, existing templates, and cyberattacks on OT systems. This article shows those threats classification into STRIDE categories implemented in their MTMT template.

AbuEmera et al. [1] propose, in a first instance, a smart factory components catalog. Then, from this catalog, authors build a generic smart factory model in MTMT. After that, authors have applied Khan et al.'s method [12]. Then, for identified threats, authors provide threat rules which can be integrated in MTMT and a threat assessment by scoring threats severity through CVSS v3.1 (Common Vulnerability Scoring System) representation. Finally, authors propose recommendations for security of smart manufacturing systems.

3 USE CASE: AN INDUSTRIAL PLC ARCHITECTURE

In this section, we will describe a use case which will serve to present our proposed development and results. Based on our knowledge, there are 3 main types of architectures in industrial control systems (ICS). The smallest one, PLC architecture, control a single physical process. The PLC (Programmable Logic Controller) device is the center of these architectures, it collects the data from sensors, transfers them to the HMI (Human-Machine Interface) and sends commands to actuators. Then, distributed control system (DCS) architectures control several physical processes in the same plant. Finally, supervisory control and data acquisition (SCADA) architectures are usually defined as wide range DCS that control several plants.

Our use case, presented in Figure 1, is a typical industrial PLC architecture. We have chosen a simple use case to focus on the description of our solutions. We will see later that our contributions allow to easily extend a database to include DCS and SCADA architectures.

On the top of the Figure 1, we can see a AVEVA system platform HMI software. Such HMI allows an operator to supervise the system. It displays plant information to the operating personnel graphically including a schematic representation of the plant, alarms and events logging. Underneath the HMI, the architecture includes a Yokogawa Centum VP PLC. In this use case, the PLC is used to control the system. It gathers data from the physical process and acts according to its embedded logic. Finally, connected to the Centum VP PLC, are two Honeywell ControlEdge RTUs (Remote Terminal Unit). RTUs are operated as ADC (Analog-Digital Converter), that is, they digitize physical quantities from the process and transfer it to the PLC. The field components, composed of sensors and actuators, are not taken into account in this work because they are directly connected to the RTU and therefore only physically attackable.

4 DEDICATED ICS TEMPLATES BUILDING TOOL

In this section, we firstly introduce an incremental database of ICS components to model specific industrial architectures and threats. Then, in a second time, we present an automated tool to generate MTMT templates from the database. These contributions allow to generate a template with a wide range of components in order to model various specific ICS in MTMT.

4.1 ICS template

The first step in MTMT is to model the system's architecture through a data-flow diagram. To perform this stage, the modeler requires a template. MTMT does not provide any template for ICS. AbuEmera et al. [1] proposed a handmade MTMT template with a list of threats manually created. As their approach is made of a fixed list of generic devices and threats (e.g., PLCs, RTUs, etc.), no automated method can easily allow to consider real devices and their specific threats as part of their template. Thus, we created a new template oriented to automated extraction of threats from vulnerability databases allowing for tailored risk identification.

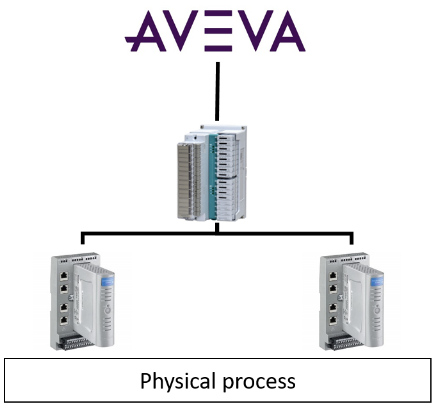

Figure 1: Use case architecture

As mentioned in Section 2.2, a MTMT template is composed of stencils and threat types. Stencils describe generic components, such as *Communication Protocol* or *PLC*, which are declined in specific components named derived stencils, as *ModBus TCP Protocol* for *Communication Protocol* or *Centum VP* for *PLC*. Stencils are described by their properties. For example, *Communication Protocol* has a physical medium, can (or cannot) provide source authentication, etc. These properties are specific to a stencil and are inherited by each derived stencils. Then, we can define a specific component (derived stencil) from a generic one by fixing one or several properties value as a constraint. In our use case, *ModBus TCP* does not provide source authentication. Then, threat types are structured in threat families (spoofing, tampering, etc.) named Categories and declined in threat types (e.g. Spoofing the source device, spoofing the destination device, etc.). Each threat type is associated with stencils through rules (e.g., spoofing a source is not possible if the protocol authenticates the source). Examples of stencils, derived stencils, categories, and threat types are provided in Figure 2.

We define, in our MTMT ICS template, stencils which are representative of standard components found in a PLC architecture as presented in Section 3. Moreover, we add to the Generic Data Flow stencil, a preconfigured common industrial communication protocol of this type of architecture, ModBus TCP. However, we argue that only considering generic devices or protocols (e.g., PLC) as previously done by FLa et al. [7] or AbuEmera et al. [1] is not precise enough. Indeed, some industrial equipment are more vulnerable than others and it is hardly possible to state that for instance "A generic PLC would have a threat regarding tampering". To circumvent this limitation, we propose to add real off-the-shelf equipment to the template as derived stencils. For that, we defined a set of configurable properties to each generic stencil. These properties

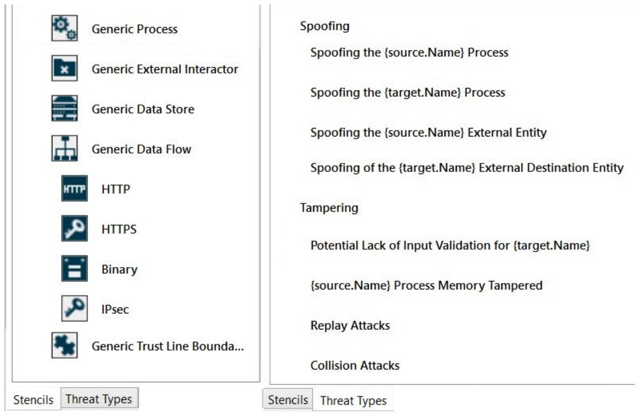
Generic Process
Generic External Interactor
Generic Data Store
Generic Data Flow
HTTP
HTTPS
Binary
IPsec
Generic Trust Line Bounda...
Stencils
Threat Types
Spoofing
Spoofing the {source.Name} Process
Spoofing the {target.Name} Process
Spoofing the {source.Name} External Entity
Spoofing of the {target.Name} External Destination Entity
Tampering
Potential Lack of Input Validation for {target.Name}
{source.Name} Process Memory Tampered
Replay Attacks
Collision Attacks
Stencils
Threat Types

Figure 2: Example of stencils, derived stencils, categories, and threat types

are inherited by derived stencils. So, to define a specific equipment, the modeler sets values in accordance with the characteristics of the device or data flow. For example, in our use case we defined 4 generic stencils (HMI, RTU, Data flow, and PLC) for which we customized derived stencils for each device/software of our use case, presented in Figure 1, respectively AVEVA system platform, ControlEdge RTU, ModBus TCP, and Centum VP.

Therefore, the quality of the model depends on the relevance of our properties. In the default Microsoft template, stencils already exist to define software (Generic Process) or communication protocols (Data Flows). So, we have defined properties of the AVEVA system platform and ModBus TCP derived stencils according to our knowledge and the publicly available documentation.

To construct PLC and RTU stencils and properties we used protection profiles [3] for PLC provided by ANSSI, the French cybersecurity agency. This protection profile was written as part of the work of the Industrial Systems Cybersecurity Working Group (ISWG), which includes companies such as Phoenix Contact, Schneider Electric, Thales or Siemens.

Protection profiles identify PLC parameters (sensitive assets, their threats, and their associated security objectives) which fit well with MTMT structure. Indeed, MTMT defines threats according to configurable properties (security objectives of a PLC sensitive assets) inherit from a stencil (Generic PLC) which values are set for each derived stencil (specific PLC as Centum VP). So, we mapped security objectives of sensitive assets to properties. For example, the sensitive asset "Firmware" have to be protected in Integrity and Authenticity. We therefore created a configurable property named "Firmware" for the PLC stencil, which permits to specify if the derived stencil (as Centum VP) firmware is protected in integrity and authenticated. Then we created a threat type rule which matches the lack of firmware Integrity or Authenticity to its threat "Firmware corruption". The entire list of sensitive assets, threats, and security objectives is available in the protection profile file.

We consider that industrial devices, as RTU, may be viewed as PLC with less functionalities. For instance, properties as "Firmware" or "Operating mode" presented in Listing 1 are applicable for RTU but "User program", present in the PLC protection profile, is not a functionality provided by a RTU. This work to derive properties from PLC to other devices must be done by an expert to properly define the properties of the equipment.

In this part, we introduced how we built the skeleton of our incremental database of ICS components from a typical industrial PLC architecture and how we defined components properties from protection profiles of the French cybersecurity agency. In the following part, we present an automated tool to generate MTMT templates from the database.

4.2 Template automation for MTMT

Yet, there are currently hundreds if not thousands of industrial devices on the market and providing a template including all of them would be a tedious task. This way, we introduce a tool which allows to automatically add specific devices and their threats in MTMT as derived stencils. This tool lets any modeler specify exactly which devices are in use in their systems and have them automatically added to the template with their threats.

In section 4.1, we presented a template for ICS, made of generic devices. Our second contribution is to provide a tool that automates the creation of a template from a configuration file. The template provided in section 4.1 integrates generic devices and data flows of a PLC architecture (PLC, RTU, industrial data flow and industrial software). Based on this model, we propose a tool that generates, from a configuration file, a model with real devices needed by the designer (for example, a Yokogawa Centum VP PLC instead of a generic PLC). This configuration file will work as a database of real devices, data flows and threats. We can see this configuration file as a community database that modelers increment according to the devices they need. Once a device is created, it will be accessible to all users. This tool also includes the research of known vulnerabilities (describe later in Section 5) for all devices mentioned in the configuration file.

We chose to use TOML [17] (Tom's Obvious, Minimal Language) for the configuration file. TOML is a configuration file format designed to be easily read and written with an open source specification. It is structured through three main items : sections ([section.sub-section]), pairs (key = value), and comments (# comment). Listing 1 introduces a partial configuration file of the use case already filled with the information of our generic ICS template. Modelers only need to add which devices are required for their system and for which a research of CVE will be carried out. We choose to define TOML file as database instead of the XML-based MTMT template for the following reasons. Firstly, a model is associated with a single template, and thus, incrementing the template creates a conflict with former models. Secondly, TOML configuration file can be easily used for other tools.

We propose the ability to filter CVE according to a period of time (e.g., since 2001, between 2010 and 2020, etc.) and/or a severity metric (CVSS score). Indeed, such filtering allows to avoid numerous vulnerabilities that do not matter (e.g., too old or not critical). CVE are collected on NVD vulnerability database [16] according to the configuration file (devices and filters). Without filters there are 7 vulnerabilities for the AVEVA system platform, 25 for Yokogawa Centum VP, and 3 for Honeywell ControlEdge RTU (CVE research with incomplete names can lead to false positives). With filters defined in the TOML file, presented in Listing 1, we decrease vulnerabilities to 14 for Yokogawa Centum VP (11 less).

Obviously, the default ICS template can be customized by users for their particular needs adding threats from their CTI (Cyber Threat Intelligence), stencils for their domain-specific device family, etc.

Listing 1: Partial configuration file of the ICS template

```
[Stencil_search_CVE]
        [Stencil_search_CVE.1]
        name = "CENTUM VP"
        period = ["2021", "2022", "2023"]
        severity = ["HIGH", "CRITICAL"]
        [Stencil_search_CVE.2]
        name = "AVEVA system platform"
        [Stencil_search_CVE.3]
        name = "ControlEdge RTU"

[stencils]
        [stencils.2]
        name = "PLC"
        description = "PLC"
        img_loc = "ImageMachineTrustBoundaryArc7.png"
        Repre_stencil = "t_e"

                [stencils.2.proprieties1]
                name = "Firmware signature"
                values1 = "Not Selected"
                values2 = "Yes"
                values3 = "No"

                [stencils.2.proprieties2]
                name = "Malformed input management"
                values1 = "Not Selected"
                values2 = "Yes"
                values3 = "No"
        ...
[derived_stencils]
        [derived_stencils.101]
        stencil_base = "PLC"
        name = "CENTUM VP"
        description = "Yokogawa Electric Corporation CENTUM
            VP"
        img_loc = "ImageMachineTrustBoundaryArc7.png"

        [derived_stencils.102]
        stencil_base = "PLC"
        name = "SOFREL S4W"
        description = "Lacroix SOFREL S4W "
        img_loc = "ImageMachineTrustBoundaryArc7.png"
        ...
[Threat]
    [Basic_Type_Threat.214]
        Parent = "Tampering"
        name = "Firmware alteration"
        include = "target is [PLC]"
        exclude = "target.[Firmware signature] is 'Yes'"
        Description = "The attacker manages to inject and
             run a corrupted firmware on the {target.name}.
             The code injection may be temporary [...].
             Finally, the attacker manages to modify the
             version of the firmware installed on the {
             target.name} without having the privilege to do
              so"

        [Basic_Type_Threat.215]
        Parent = "Tampering"
        name = "Execution mode alteration"
        include = "target is [PLC]"
        exclude = "target.[Integrity and authenticity of
             execution mode] is 'Yes' or target.[Integrity
             and authenticity of execution mode] is 'Provide
              integrity'"
        Description = "The attacker manages to modify the
             execution mode of the ToE without being
             authorized (a stop command for instance)."
```

Once the configuration file is filled, our tool automatizes the creation of the corresponding XML-based MTMT template. First, the tool retrieves all stencils defined in the TOML file and inserts those into the template. Then, all derived stencils are in turn inserted into their respective stencil.

Secondly, threats are added to the template. As stencils, threats categories are inserted before derived ones. Threat categories are the ICS (as mentioned in Section 4.2) and those provided by default in MTMT, that is, STRIDE categories and abuses (legitimate user who violates the terms of use for the system without violating a system security policy). We define 2 kinds of derived threats : **c**lassical threats and **C**VE-based threats. Classical threats are those created by a standard usage of MTMT applying STRIDE methodology. For example, the threat "*spoofing source*" indicates that no authentication mechanism is provided by the source of the interaction. CVE-based threats are those retrieved by our tool for devices specified in the configuration file. Figure 3 shows the MTMT use case model built from our template.

Our ICS template is available online[1] including the TOML configuration file, the automation code, and a *readme* file allowing to reproduce the method. The code, written in Python, is a proof-of-concept and has limitations. For the moment, we are not able to define properties for derived stencils and to set a predefined value for derived stencils constrains from the configuration file.

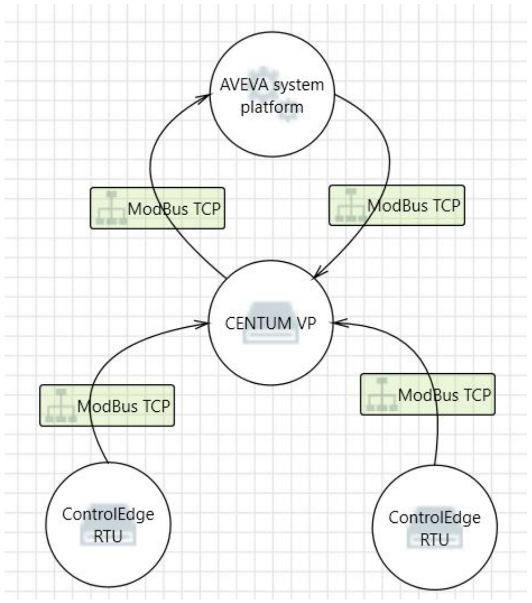

Figure 3: Use case MTMT model

[1]https://github.com/StrideICS/StrideICS

Table 1: Mapping CWE with STRIDE

STRIDE	CWE/Technical Impact	CWE/Scope
Spoofing	Gain privileges / Assume identity	Access control - Authentication
Tampering	Modify data	Integrity
Repudiation	Hide activities	Non-Repudiation - Accountability
Information disclosure (privacy breach or data leak)	Read data	Confidentiality
Denial of service	DoS: unreliable execution	Availability
Denial of service	DoS: resource consumption	Availability
Elevation of privilege	Execute unauthorized code or commands	Confidentiality - Integrity - Availability - Access control
Elevation of privilege	Bypass protection mechanism	Access Control - Authentication

5 CVE MODELING IN MTMT

This section aims to explain our third contribution that allows to integrate CVE in MTMT. Considering that MTMT uses STRIDE for the threat modeling, our goal is to match CVE with STRIDE categories.

Honkaranta et al. [8] provide a mapping table, depicted in Table 1, between CWE and STRIDE using CWE's *Scope* and *Technical Impact* properties which are defined as follows.

- **CWE Scope:** Identifies the application's security area that is violated;
- **CWE Technical Impact:** Describes the negative technical impact if an adversary succeeds in exploiting this weakness.

To properly understand the Table 1, it is necessary to remind here that a cyberattack consists in the realization of a threat by the exploitation of a vulnerability which causes damage. The mapping made in Table 1 between "Gain privileges" and Spoofing must be read as follows. An attacker can use a lack of authentication or access control (CWE Scope) to spoof something or someone (Threat) in order to gain privileges or assume an identity (CWE Scope). Indeed, in this context Elevation of privilege and Spoofing have a close meaning but there are still distinguishable by their impact.

As mentioned by Honkaranta et al., the NIST Vulnerability Database (NVD) provides links from CVE to CWE and gives the possibility to match CVE with STRIDE. However, Loveless [15] showed in 2008 the difficulty of mapping CVE to a CWE correctly. According to Lovelss, up to 25% (missing, need details, inclusion) of CVE may not have corresponding CWE and consequently Honkaranta et al.'s method is inapplicable. To fill this gap, numerous articles [6, 11, 20, 21] propose AI-based solutions to map CVE to CWE. However, none of them made their program, algorithm, or source code available. To overcome this problem, we propose an improved method to map CVE with CWE in this article.

For CVE which have no CWE attribution, we propose to match them with STRIDE categories through the use of CVSS (Common Vulnerability Scoring System) v3.1 score. CVSS score contain several metrics allowing to estimate impacts on confidentiality, integrity, and availability evaluated as None, Low, or High. Confidentiality, integrity, and availability correspond to the mitigation, respectively, of information disclosure, tampering, and denial of service threats from STRIDE. So, we match CVE without corresponding CWE by matching CVSS score with STRIDE in the following way. If an impact metric is different from None we associate the CVE with the corresponding STRIDE threat. Yet, CVSS score does not integrate Spoofing, Repudiation, and Elevation of privilege threats and can lead to a partial mapping. For example, WinCC CVE-2021-27384 "A vulnerability has been identified in SIMATIC HMI Comfort Outdoor Panels [...]. SmartVNC has an out-of-bounds memory access vulnerability in the device layout handler, represented by a binary data stream on the client side, which can potentially result in code execution." has an high impact on confidentiality, Integrity, and Availability in CVSS score vector. Our method will not take into account the Elevation of privilege threat (unauthorized code execution). Even if our method (compared to an intermediate mapping with CWE) tends to merge threats together, we still identify that a threat exists and must be addressed. Moreover, we argue that often, adding a countermeasure for one security objective will also improve other objectives (e.g., adding authentication also often prevents tampering, or information disclosure).

6 CONCLUSION AND PERSPECTIVES

In this paper, we presented the context of industrial cybersecurity and the need for dedicated automated tools. To this end, we proposed a configuration file used to build a database of industrial communication protocols and devices which can be customized and incremented by modelers. Then, we presented our tool which automatically generates a MTMT template based on the data flows and devices defined in the configuration file. In addition, we explained our methodology for gathering component CVEs and matching them to STRIDE and thus integrating the CVEs within MTMT. These tools allow companies to quickly build a large database of industrial components to efficiently design threat models dedicated to their project and thus better integrate cybersecurity into their systems. As a side note, we think that our contributions could fit in other fields than industrial systems. However, as ICS are our primary focus, we left the validation of this statement for future work and presently claim our contributions mainly regarding ICS.

In a future work, we plan to further improve the filtering of CVE based on stencil characteristics, such as the firmware version of the device, the version of the software or the configuration parameters like the activation or deactivation of some network protocols. Moreover, we would like to provide a mapping method between protection profile and STRIDE. Finally, we will extend our ICS template with the other ten protection profiles provided by ANSSI to permit model more complex industrial architectures as DCS or SCADA.

ACKNOWLEDGMENTS

This work was supported by the French National Research Agency in the framework of the "Investissements d'avenir" program (IRT Nanoelec, ANR-10-AIRT-05)

REFERENCES

[1] Eman A AbuEmera, Hesham A ElZouka, and Amani A Saad. 2022. Security Framework for Identifying threats in Smart Manufacturing Systems Using STRIDE Approach. In *2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE)*. IEEE, 605–612.

[2] Md Rashid Al Asif, Khondokar Fida Hasan, Md Zahidul Islam, and Rahamatullah Khondoker. 2021. STRIDE-based Cyber Security Threat Modeling for IoT-enabled Precision Agriculture Systems. In *2021 3rd International Conference on Sustainable Technologies for Industry 4.0 (STI)*. IEEE, 1–6.

[3] ANSSI. 2015. *Protection profiles for industrial systems*. https://www.ssi.gouv.fr/guide/profils-de-protection-pour-les-systemes-industriels/ [Online; accessed 10-January-2023].

[4] ANSSI. 2019. *EBIOS Risk Manager*. https://www.ssi.gouv.fr/en/guide/ebios-risk-manager-the-method/ [Online; accessed 10-January-2023].

[5] Microsoft Corporation. 2020. *Microsoft Threat Modeling Tool*. https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling [Online; accessed 10-January-2023].

[6] Siddhartha Shankar Das, Edoardo Serra, Mahantesh Halappanavar, Alex Pothen, and Ehab Al-Shaer. 2021. V2w-bert: A framework for effective hierarchical multiclass classification of software vulnerabilities. In *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 1–12.

[7] Lars Halvdan Flå, Ravishankar Borgaonkar, Inger Anne Tøndel, and Martin Gilje Jaatun. 2021. Tool-assisted threat modeling for smart grid cyber security. In *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*. IEEE, 1–8.

[8] Anne Honkaranta, Tiina Leppänen, and Andrei Costin. 2021. Towards practical cybersecurity mapping of stride and cwe—a multi-perspective approach. In *2021 29th Conference of Open Innovations Association (FRUCT)*. IEEE, 150–159.

[9] KU Leuven imec DistriNet. 2020. *LINDDUN privacy threat modeling framework*. https://www.linddun.org/ [Online; accessed 10-January-2023].

[10] Joseph Ingeno. 2018. *Software Architect's Handbook: Become a Successful Software Architect by Implementing Effective Architecture Concepts*. Packt Publishing.

[11] Konstantin Izrailov, Mikhail Buinevich, Igor Kotenko, and Alexander Yaroshenko. 2021. Identifying characteristics of software vulnerabilities by their textual description using machine learning. In *2021 World Automation Congress (WAC)*. IEEE, 186–192.

[12] Rafiullah Khan, Kieran McLaughlin, David Laverty, and Sakir Sezer. 2017. STRIDE-based threat modeling for cyber-physical systems. In *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. IEEE, 1–6.

[13] Loren Kohnfelder and Garg Praerit. 1999. *The threats to our products, Microsoft Interface*. https://web.archive.org/web/20100117125404/http://blogs.msdn.com/sdl/attachment/9887486.ashx

[14] Ralph Langner. 2011. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy* 9, 3 (2011), 49–51.

[15] M Loveless. 2008. *CWE mapping analysis*. https://cwe.mitre.org/documents/mapping_analysis/index.html [Online; accessed 10-January-2023].

[16] NIST. 2021. *National Vulnerability Database*. https://nvd.nist.gov/ [Online; accessed 12-January-2023].

[17] Tom Preston-Werner. 2022. *Tom's Obvious Minimal Language*. https://toml.io/en/ [Online; accessed 12-January-2023].

[18] Adam Shostack. 2014. *Threat Modeling: Designing for Security* (1st ed.). Wiley Publishing.

[19] VerSprite. 2021. *PASTA Threat Modeling*. https://versprite.com/blog/what-is-pasta-threat-modeling/ [Online; accessed 10-January-2023].

[20] Tianyi Wang, Shengzhi Qin, and Kam Pui Chow. 2021. Towards Vulnerability Types Classification Using Pure Self-Attention: A Common Weakness Enumeration Based Approach. In *2021 IEEE 24th International Conference on Computational Science and Engineering (CSE)*. IEEE, 146–153.

[21] Tao Wen, Yuqing Zhang, Ying Dong, and Gang Yang. 2015. A Novel Automatic Severity Vulnerability Assessment Framework. *J. Commun.* 10, 5 (2015), 320–329.