SPECIFICATION

Submodel Vulnerability Managemnet

Version 1

21.06.2023

Submodel Template of the

Asset Adminstration Shell

Imprint

Publisher

Steinbeis Innovation gGmbH
Adornostr. 8
70599 Stuttgart
Germany

Source for Specification Document

Plattform Industrie 4.0
Bertolt-Brecht-Platz 3
10117 Berlin
Germany

Authors

Frank Scherenschlich
Christian Albert
Sophia Machaidze

Version history

2023-08-28	1.0	Submission to the InterOpera Consortium

Content

List of Figures

List of Tables

Foreword

This document is the result of the InterOpera project “Vulnerability Management”. The development of the submodel was coordinated by the company Class.Ing as project manager. People from different companies took part in the conception in different roles, which considered many of the market requirements. Participants of the workshops and the elaboration were:

Frank Scherenschlich, Class.Ing Ingenieur-Partnerschaft

Christian Albert, Class.Ing Ingenieur-Partnerschaft

Sophia Machaidze, Class.Ing Ingenieur-Partnerschaft

Andre Bröring, Technische Hochschule Ostwestfalen-Lippe

Andreas Harner, VDE Verband der Elektrotechnik Elektronik Informationstechnik e. V.

Dr. Markus Schoisswohl, Hegla New Technology Gmbh & Co KG

Stefan Ullmann, ifm electronic GmbH

Kai Schwermann, bill-X GmbH

Yannis Vierkötter, bill-X GmbH

We thank for the commitment.

1 General

1.1 About this document

This document is a part of a specification series. Each part specifies the contents of a Submodel template for the Asset Administration Shell (AAS). The AAS is described in [1-3] and [6]. First exemplary Submodel contents were described in [4], while the actual format of this document was derived by the "Administration Shell in Practice" [5]. The format aims to be very concise, giving only minimal necessary information for applying a Submodel template, while leaving deeper descriptions and specification of concepts, structures and mapping to the respective documents [1-6].

The target group of the specification are developers and editors of technical documentation and manufacturer information, which are describing assets in smart manufacturing by means of the Asset Administration Shell (AAS) and therefore need to create a Submodel instance with a hierarchy of SubmodelElements. This document especially details on the question, which SubmodelElements with which semantic identification shall be used for this purpose.

1.2 Scope of the Submodel

This Submodel template aims at interoperable provision of information describing Vulnerability Management in regard to the asset of the respective Asset Administration Shell. Central element is the provision of properties [7], ideally interoperable by the means of dictionaries such as ECLASS and IEC CDD (Common Data Dictionary). The purpose of this document is to make selected specifications of Submodels in such manner that information about assets can be exchanged in a meaningful way between partners in a value creation network. It targets equipment for process industry and factory automation by defining standardized meta data].

The intended use-case is the provision of a standardized property structure for Vulnerability Management, which enables the standardization of vulnerabilities. The more detailed description can be found in chapter 2.

This concept can serve as a basis for standardizing the respective Submodel. The conception is based on existing norms, studies of common practices at enterprises, directives, and standards so that a far-reaching acceptance can be achieved.

Beside standardized Submodel this template also introduces standardized SubmodelElementCollections (SMC) to improve the interoperability while modelling aspects of Software Bill of Material within other Submodels.

1.3 Relevant standards and sources of concepts for the Submodel template

According to [3], interoperable properties might be defined by standards, consortium specifications or manufacturer specifications. Useful standards providing sources of concepts are:

Table 1: List of examplary standards defining interoperable properties

DIN EN 62559-2	Use case methodology Part 2: Definition of the template for use

	cases, actor list and requirements list

So called property dictionaries are used identify information elements (see Terms and Definitions of [6]). Such property dictionaries include:

- ECLASS, see: https://www.eclasscontent.com/
- IEC CDD, see: https://cdd.iec.ch/cdd/iec61987/iec61987.nsf and https://cdd.iec.ch/cdd/iec62683/cdddev.nsf

In this document, properties are aimed to be described by ECLASS.

2 Information set for Submodel Contact Information

2.1 Brief description

Hardware and / or software components as well as products resulting from them may have vulnerabilities that are documented and provided by the respective manufacturer / provider. In the context of the use of the components or products, these vulnerabilities are required by the user to determine the relevance for his products, to evaluate them and to pass them on if necessary. Naturally, these vulnerabilities have a close version related link to the bill of material (BOM) or software bill of material (SBOM) in order to establish a clear reference to the source.

The definition of vulnerabilities is either done by a provider / manufacturer itself in the form of advisories and / or standardised in central vulnerability databases. Standardised formats are used for the description and exchange. The description of vulnerabilities, the reference to existing data sources and the assessment of vulnerabilities is mapped in a role dependent submodel in the administration shell (AAS).

The AAS submodel "Vulnerability Management" represents a formal, machine readable inventory of vulnerabilities with a reference to the software and hardware BOM and to publicly available descriptions. In the administration shell, the individual elements are described on a meta-level. The currently existing standardised formats for the interoperable exchange of vulnerabilities are taken into account.

2.2 Description of the "Vulnerability Management" submodel

The "Vulnerability Management" submodel contains not only the description of the vulnerabilities at meta-level, but also the reference to the publicly disclosed vulnerability in advisories or vulnerability databases. It also has references to the vulnerability relevant hardware and / or software. The meta data fields were determined from the CSAF standard. They generally represent the most important attributes for describing entries (differentiated according to optional and mandatory characteristics). The actual description of the vulnerabilities then takes place in this format and is also transferred as a file, which ensures machine-readable further processing. In the event of a change to the CSAF standard, no adaptation of the submodel is necessary.

2.3 Focus

The submodel "Vulnerability Management" focuses on the mapping of vulnerabilities with referencing to corresponding version-dependent parts lists. The actual component or the actual product is described via a submodel "Nameplate". Other submodels, which also provide version- and / or element dependent specifications, are addressed via references. This provides a clear separation of the contents.

2.4 Process representation and role consideration

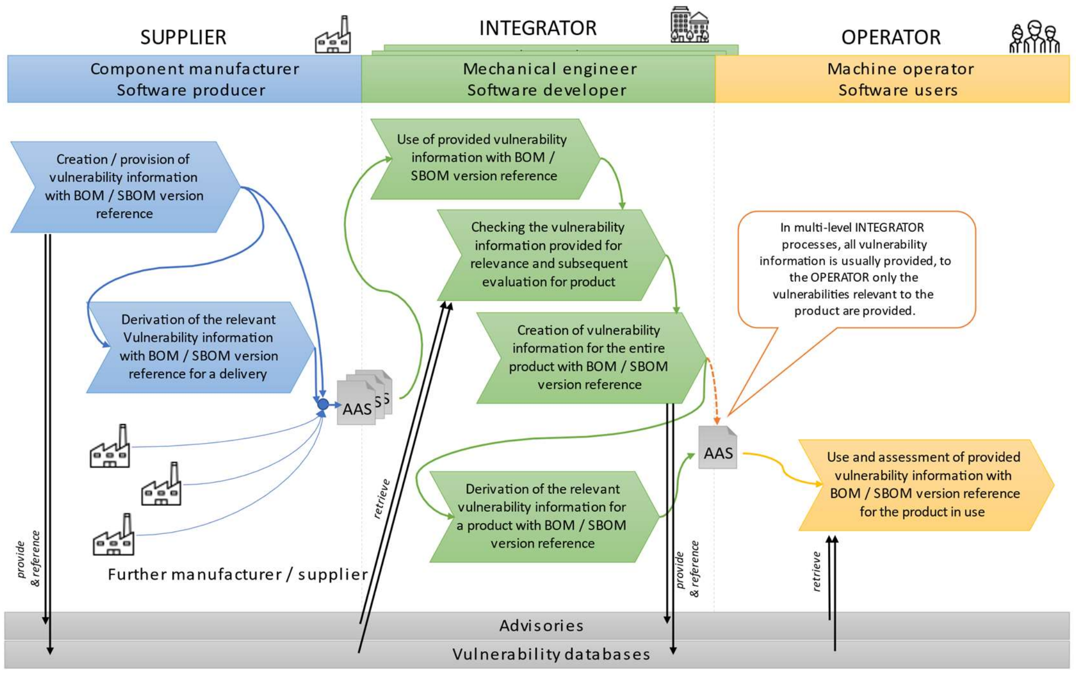
SUPPLIER
Component manufacturer
Software producer
INTEGRATOR
Mechanical engineer
Software developer
OPERATOR
Machine operator
Software users
Creation / provision of vulnerability information with BOM / SBOM version reference
Derivation of the relevant Vulnerability information with BOM / SBOM version reference for a delivery
AAS
Further manufacturer / supplier
provide & reference
retrieve
Use of provided vulnerability information with BOM / SBOM version reference
Checking the vulnerability information provided for relevance and subsequent evaluation for product
Creation of vulnerability information for the entire product with BOM / SBOM version reference
Derivation of the relevant vulnerability information for a product with BOM / SBOM version reference
In multi-level INTEGRATOR processes, all vulnerability information is usually provided, to the OPERATOR only the vulnerabilities relevant to the product are provided.
Use and assessment of provided vulnerability information with BOM / SBOM version reference for the product in use
Advisories
Vulnerability databases

Figure 1: Process illustration Vulnerability Management

The figure shows two different data flows. The coloured arrows represent the data flow through the administration shells and the black arrows represent the data flow via advisories and / or vulnerability databases.

2.5 Actors

Table 2: Actors

File name	Actor type	Description Actor	Tasks
SUPPLIER	Business	Manufacturers of hardware and / or software components provided at the beginning of the value chain	• Version-dependent description of the developed hardware and / or software components using standard formats in the form of bills of materials (BOM, SBOM) as a basis for the assignment of vulnerabilities • Creation of vulnerability information in own advisories and / or provision of vulnerability information in vulnerability databases such as NIST NVD or CERT@VDE • Use of CSAF as standard description of vulnerability elements • Referencing the hardware and / or software in the vulnerabilities

			• Instance-based derivation of the relevant vulnerability information for delivered components • Provision of the vulnerability information in the administration shell AAS in typed form (all versions of the component) or in instance form (reduction to the component provided). In order for this concept to function optimally and efficiently, all SUPPLIERS provide their vulnerability information to the INTEGRATOR in the form of management trays.
INTEGRATOR	Business	Machine builders / software developers who use components from SUPPLIER to create their products and who develop their own products as part of the value chain and make them available to the market.	• Use of provided component vulnerability information of different SUPPLIER and integration into the documentation of the own development depending on the component version ○ Checking the vulnerability information provided for relevance to the own products created and evaluation for possible remediation of the vulnerabilities • Description of own vulnerabilities with reference to the used component in the same version-dependent manner, if the SUPPLIER does not provide corresponding machine-readable data. • Creation of vulnerability information either as an advisory and / or in a vulnerability database in the context of publication ○ with interaction to the corresponding bill of material (BOM, SBOM) • Instance-based derivation of relevant vulnerability information for delivered products ("As-Built ") Often the role of the INTEGRATOR is a multi-stage process. In this case, the usual n-1 | n | n+1 mapping is used, which also maps the entire "process of refinement".

OPERATOR	Business	User of the hardware and / or software provided by INTEGRATOR	• Use of the instance vulnerability information provided by the INTEGRATOR • Assessment of the vulnerability (information) in the context of the own operating environment for relevance ("As-IS") • Commenting on the relevance in VEX entries on the vulnerability • Elimination of vulnerabilities in the own operating environment and documentation

2.6 Interaction of the roles

The following consideration is based on an analysis of the roles, their respective tasks and the interaction between the roles.

The roles represent independent actors according to the process view. This means that data is exchanged between these actors in a standardised form:

- SUPPLIER provide machine-readable data for the INTEGRATOR
- The INTEGRATOR provides machine-readable data for the OPERATOR

The machine-readable data is meta-data describing the vulnerabilities. The actual detailed vulnerability description is usually done in advisories or vulnerability databases that are referenced.

2.7 Benefits

The consistent use of the same machine-readable formats and the further use of the data along the value chain provides a great advantage. This is done by relying on already existing and established standard formats that are integrated at the meta level. This approach enables an assignment between concrete assets and vulnerabilities.

2.8 Technical description

In the technical description, the procedural requirements are transferred to the administration shell (AAS) technology used. The submodels used are discussed as well as the type of administration shell, differentiated according to type and instance as well as "As-Built" and " As -Is".

2.9 Administration shells and submodels

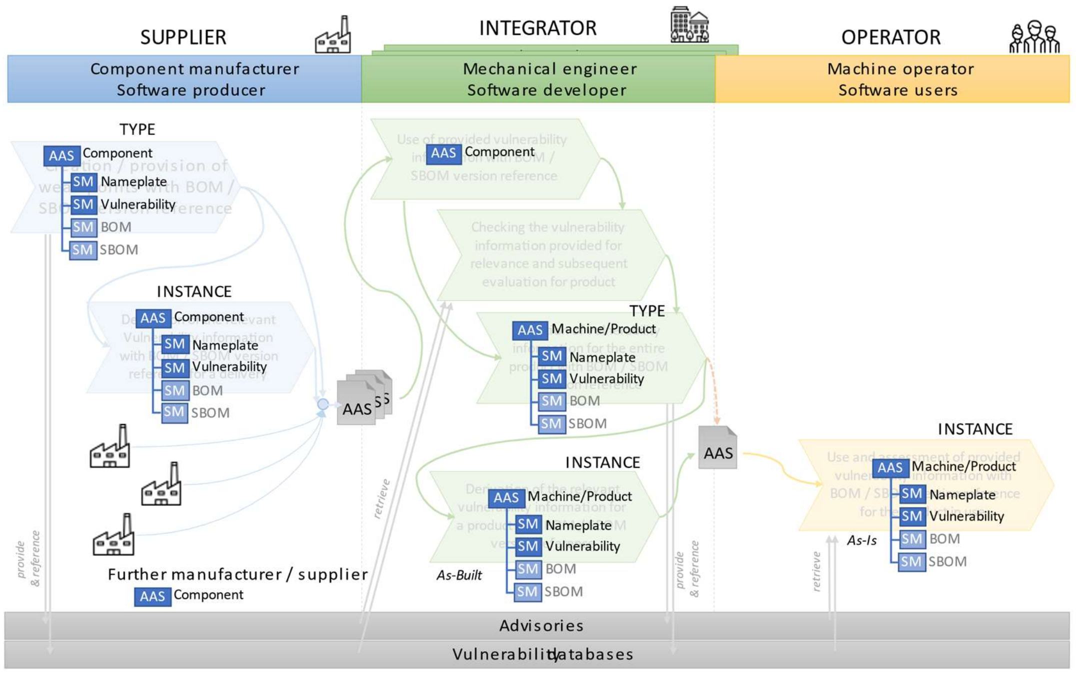

Figure 2: Technical description with the management trays

SUPPLIER

Each vulnerability is described via an entry in the "Vulnerability Management" submodel in the component's administration shell. In this way, several vulnerabilities can be assigned to one component.

The component itself is described fundamentally via a corresponding nameplate. For hardware, the submodel "Digital Nameplate" is used, for software the submodel "Software Nameplate".

The hardware and / or software used is described via the submodels "Bill of Material" (BOM) and / or "Software Bill of Material" (SBOM) depending on the version. The vulnerabilities described in the "Vulnerability Management" submodel refer to the corresponding versions of the bill of material. In the type description, all relevant vulnerabilities of the component are listed and in a possible instance description, only the respective vulnerabilities that are relevant for the provided component are listed.

For the subsequent role INTEGRATOR, ideally several SUPPLIER provide an administration shell that can then be used.

INTEGRATOR

The INTEGRATOR uses the SUPPLIER administration shells with vulnerability descriptions in the "Vulnerability Management" submodel as part of its development. This means that the vulnerabilities for the components used are available without media discontinuity. If SUPPLIERs provide no or incomplete vulnerability descriptions, the INTEGRATOR determines the vulnerabilities for these components and creates a separate administration shell AAS for these components and references it. Notifying the SUPPLIER of the vulnerability is additionally recommended in any case, but is not part of the process and model described here. Thus, all vulnerability descriptions are available for the components used (and upstream development stages of the INTEGRATOR).

The INTEGRATOR creates a separate administration shell for the product he creates, e.g. the machine or software he creates. This is fundamentally specified via a "Nameplate" and also uses the submodels "Bill of Material" (BOM) and / or "Software Bill of Material" (SBOM) for part specification. The latter then contains a list of the product's vulnerabilities in the "Vulnerability Management" submodel. The meta-level is also used, i.e. reference is made to advisories and / or entries in vulnerability databases. Usually, a type administration shell is created here, which contains all versions of the hardware and / or software with assignment of vulnerabilities.

For the transfer to the OPERATOR, i.e. the user of the product, the INTEGRATOR creates an instance administration shell that contains only the submodel contents that are also contained in the transferred version of the product. This is an "As-Built" mapping, i.e. a mapping as created and provided by the INTEGRATOR (whereby the INTEGRATOR has no digital knowledge of the deployment to the OPERATOR).

OPERATOR

The OPERATOR receives the "As-Built" instance administration shell from the INTEGRATOR, which initially corresponds to the status of the transfer and is subsequently updated via updates. With the transfer and use at the OPERATOR, it becomes an "As-IS" administration shell, as it is considered and used according to the intended use at the OPERATOR. The OPERATOR is independently responsible for managing the administration shell for the product used.

If the INTEGRATOR provides updates on vulnerabilities for the software / product used, these are transferred by the OPERATOR to the "As-IS" instance administration shell and checked and assessed there. An assessment of the vulnerabilities by the OPERATOR is documented via corresponding VEX entries.

2.10 Differentiation according to type and instance

The distinctions between type and instance of the administration shell have already been given in the previous sections. Type administration shells describe a created product comprehensively with all vulnerabilities and thus also with a reference to all components used (BOM, SBOM). For versions of a product that have been handed over, the provision focuses on the concrete handover, i.e. which vulnerabilities are relevant in which version.

2.11 Interactions with other systems

The "Vulnerability Management" submodel is usually considered in combination with hardware and / or software parts lists in order to establish a direct link to the version of the product or component. SUPPLIER and / or INTEGRATORs provide vulnerability descriptions via their own advisories or via centrally managed vulnerability databases. These sources can be referenced via the "Vulnerability Management" submodel. In this way, the subsequent roles in the process can access standardised contents of the predecessors and comprehend the entire contents of the vulnerabilities.

2.12 Sector relevance

The submodel "Vulnerability Management" was considered from different industry perspectives in the InterOpera project of the same name and thus includes coverage of various requirements. Considered in this context were:

- Software manufacturer (SUPPLIER)
    - >> Provision of vulnerabilities to software components
- Component manufacturer Hardware or hardware with software part (SUPPLIER)
    - >> Provision of vulnerabilities to hardware related components
- Software Developer (INTEGRATOR)
    - >> Development company for software products of various kinds
    - >> Use of existing vulnerability information and provision of own advisories
- Product manufacturer / machine builder (INTEGRATOR)
    - >> Use of vulnerability information from the component suppliers within the framework of the construction of machines or products.
    - >> Provision of own vulnerability information for the created element
- Software user (OPERATOR)
    - >> Use of provided vulnerability information in the context of the operation of software
- Product user / machine operator (OPERATOR)
    - >> Use of provided vulnerability information in the context of the operation of a product or plant

3 Submodel and Collections

3.1 Approach

Submodel defined in this document is usable in asset management shells for type and instance assets.

The figure below shows the UML-diagram defining the relevant properties which need to be set. Table 3 describes the details of the Submodel structure combined with examples.

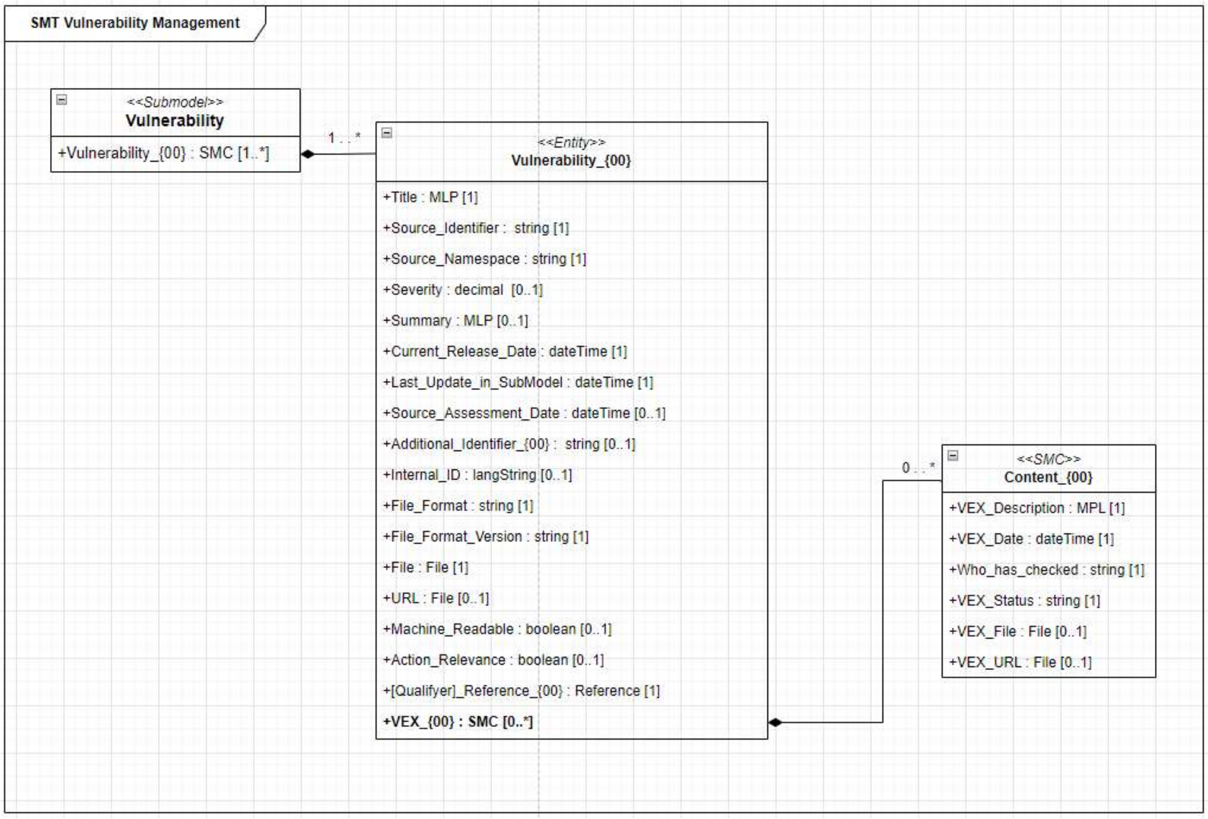
SMT Vulnerability Management
<<Submodel>>
Vulnerability
+Vulnerability_{00} : SMC [1..*]
1 . . *
<<Entity>>
Vulnerability_{00}
+Title : MLP [1]
+Source_Identifier : string [1]
+Source_Namespace : string [1]
+Severity : decimal [0..1]
+Summary : MLP [0..1]
+Current_Release_Date : dateTime [1]
+Last_Update_in_SubModel : dateTime [1]
+Source_Assessment_Date : dateTime [0..1]
+Additional_Identifier_{00} : string [0..1]
+Internal_ID : langString [0..1]
+File_Format : string [1]
+File_Format_Version : string [1]
+File : File [1]
+URL : File [0..1]
+Machine_Readable : boolean [0..1]
+Action_Relevance : boolean [0..1]
+[Qualifyer]_Reference_{00} : Reference [1]
+VEX_{00} : SMC [0..*]
0 . . *
<<SMC>>
Content_{00}
+VEX_Description : MPL [1]
+VEX_Date : dateTime [1]
+Who_has_checked : string [1]
+VEX_Status : string [1]
+VEX_File : File [0..1]
+VEX_URL : File [0..1]

Figure 3: UML-Diagram for Vulnerability Management

3.2 Properties of the Submodel “Vulnerability Management”

Table 3: Submodel “Vulnerability Management”

idShort:	**Vulnerability Management**		
Class:	**Submodel - Vulnerability**		
semanticId:	**[IRI] http://admin-shell.io/VulnerabilityManagement/1/0/Submodel/Vulnerability**		
Parent:	**AAS**		
Explanation:	**Listing of the occurring vulnerabilities of the machine**		
[SME type] idShort	**semanticId = [idType]value Description@en**	**[valueType] example**	**card.**
[SMC] Vulnerability_00	[IRI]https://admin-shell.io/VulnerabilityManagement/1/0/Vulnerability Vulnerability - A vulnerability is a weakness or error in a system, software, or process that can cause harm or disrupt normal operations	[-] 20 elements	

3.3 Properties of the Submodel Element Collection “Vulnerability_00”

Table 4: Submodel Element Collection “Vulnerability_00”

idShort:	**Vulnerability_00**		
Class:	**SubmodelElementCollection**		
semanticId:	**[IRI] https://admin-shell.io/VulnerabilityManagement/1/0/Vulnerability**		
Parent:	**Submodel - Vulnerability**		
Explanation:	**A vulnerability is a weakness or error in a system, software, or process that can cause harm or disrupt normal operations**		
[SME type] idShort	**semanticId = [idType]value Description@en**	**[valueType] example**	**card.**
[MLP] Title	[IRDI]0173-1#02-BAG165#002 Word or group of words used for identification of an object hence frequently distinguishing the object from others Title	[-] Software_Component_1_Advanced_v1@en	1
[Property] Source_Identifier	[IRDI]0173-1#02-AAW705#001 Identification	[string] ESA-2023-0001	1
[Property] Source_Namespace	[Custom]https://admin-shell.io/idta/VulnerabilityManagement/1/0/Source_Namespace Source_Namespace refers to a name or url that represents the source or origin of the vulnerability	[string] https://www.example.com/	1
[Property] Severity	[IRDI]0173-1#02-ABA749#001 extent of loss, damage or harm caused by a fault or failure	[decimal] 8,3	0..1
[MLP] Summary	[IRI]https://admin-shell.io/idta/VulnerabilityManagement/1/0/Summary A summary is a compact presentation of a content. It contains the most important information, ideas or arguments of the original and focuses on the essentials	[-] Test@de, Test@en	0..1

[Property] Current_Release_Date	[IRDI]0173-1#02-AAR969#002 quantitative expression of a point in time on a particular time scale	[dateTime] 2002-10-02T15:00:00Z	1
[Property] Last_Update_in_SubModel	[IRDI]0173-1#02-AAR969#002 quantitative expression of a point in time on a particular time scale	[dateTime] 2002-10-02T15:00:00Z	1
[Property] Source_Assessment_Date	[IRDI]0173-1#02-AAR969#002 quantitative expression of a point in time on a particular time scale	[dateTime] 2002-10-02T15:00:00Z	0..1
[Property] Additional_Identifier_00	[IRDI]0173-1#02-AAW705#001 Identification	[string] CVE-2023-0115	0..1
[Property] Internal_ID	[IRDI]0173-1#02-ABC422#001 referenceable identifier	[langString] 123abc	0..1
[Property] File_Format	[IRI]https://admin-shell.io/idta/VulnerabilityManagement/1/0/File_Format File_format describes the kind of format. A format is a fixed structure under which something is configured describes the kind of format. A format is a fixed structure under which something is configured	[string] CSAF	1
[Property] File_Format_Version	[IRI]https://admin-shell.io/idta/VulnerabilityManagement/1/0/File_Format_Version File_Format_Version refers to a specific release of a file format. It indicates the version number or identifier associated with a particular format, representing changes, updates, or revisions made to the structure and specifications of the file format over time	[string] 2.0	1
[File] File	[IRDI]0173-1#02-ABI504#001 MIME-Type, file name and file contents given by the file SubmodelElement MIME type, filename and file content specified by the SubmodelElement file	[-] /aasx/files/bsi-2022-0001.json	1
[File] URL	[IRDI]0173-1#02-ABA690#001 identifier according to automatic identification of physical objects and information on physical objects in it systems, particularly iot systems. it is used to uniquely identify a device globally and to refer to the digital nameplate	[-] https://www.example.com/.well-known/csaf/white/2023/esa-2023-0001.json	0..1
[Property] Machine_Readable	[IRI]https://admin-shell.io/VulnerabilityManagement/1/0/Machine_Readable Machine_Readable is a term that refers to the ability of documents or data to be read and processed automatically by machines	[boolean] Yes	0..1
[Property] Action_Relevance	[IRI]https://admin-shell.io/VulnerabilityManagement/1/0/Action_Relevance Action_Relevance refers to the assessment of the need or urgency to perform a given action. It involves determining whether an action is necessary to achieve a goal or to respond appropriately to a given situation	[boolean] Yes	0..1

[Ref] Reference_00_ToSBOM	[IRI]https://admin-shell.io/VulnerabilityManagement/1/0/Reference_00 References represent the modeling of logical connections	[-] (Submodel)(local)[IRI]https://example.com/ids/sm/5385_8032_5032_1790,(Entity)(local)[IdShort]Software_Machine,(Entity)(local)[IdShort]Software_Modul_A,(Entity)(local)[IdShort]Software_Component_1,(Entity)(local)[IdShort]Software_Component_1_Advanced_v1	1
[SMC] VEX_00	[IRI]http://admin-shell.io/VulnerabilityManagement/1/0/Vulnerability00/VEX_00 VEX is a vulnerability management framework and platform that focuses on the exchange of information related to vulnerabilities and their exploitability. It provides a centralized system for collecting, sharing, and analyzing data on vulnerabilities, including their severity, impact, and potential exploits. VEX helps security professionals and organizations assess and prioritize vulnerabilities, enabling effective vulnerability management and mitigation strategies	[-] 6 elements	0..*

3.4 Properties of the Submodel Element Collection “VEX_00”

Table 5: Submodel Element Collection “VEX_00”

idShort:	**VEX_00**		
Class:	**SubmodelElementCollection**		
semanticId:	**[IRI] http://admin-shell.io/VulnerabilityManagement/1/0/Vulnerability00/VEX_00**		
Parent:	**SubmodelElementCollection - Vulnerability_00**		
Explanation:	**VEX is a vulnerability management framework and platform that focuses on the exchange of information related to vulnerabilities and their exploitability. It provides a centralized system for collecting, sharing, and analyzing data on vulnerabilities, including their severity, impact, and potential exploits. VEX helps security professionals and organizations assess and prioritize vulnerabilities, enabling effective vulnerability management and mitigation strategies**		
[SME type] idShort	**semanticId = [idType]value Description@en**	**[valueType] example**	**card.**

[MLP] VEX_Description	[IRDI]0173-1#02-AAN466#002 text note	[-] Test@de, Test@en	1
[Property] VEX_Date	[IRDI]0173-1#02-AAR969#002 Date quantitative expression of a point in time on a particular time scale	[dateTime] 2002-10-02T15:00:00Z	1
[Property] Who_has_checked	[IRDI]0173-1#02-AAV999#001 Name of the person who investigated the vulnerability	[string] Max Mustermann	1
[Property] VEX_Status	[IRI]https://admin-shell.io/VulnerabilityManagement/1/0/VEX_Status Current state or situation of an object or a system	[string] affected	1
[File] VEX_File	[IRDI]0173-1#02-ABI504#001 MIME-Type, file name and file contents given by the file SubmodelElement	[-]	0..1
[File] VEX_URL	[IRDI]0173-1#02-ABA690#001 identifier according to automatic identification of physical objects and information on physical objects in it systems, particularly iot systems. it is used to uniquely identify a device globally and to refer to the digital nameplate	[-] https://nvd.nist.gov/vuln/detail/CVE-2023-0115	0..1

4 Description of the example

AAS for their components (as a type administration shell). These are available as individual AAS with the description of the nameplate the used hardware (BOM) and software (SBOM) and the vulnerabilities. The administration shells are used by the INTEGRATOR.

The INTEGRATOR builds a machine with the components supplied by the supplier:

- Component 1 with hardware and software (purchase component)
- Component 2 with hardware and software (purchase component)

The machine created from the components takes into account both the vulnerabilites of the individual components and possible vulnerabilities that have been added by the construction of the machine.

The INTEGRATOR first provides a type AAS with all versions of the machine. When the machine is transferred to the OPERATOR, the INTEGRATOR also creates an instance of the AAS which represents the "As-Built" state and is also transferred.

The OPERATOR receives the instance AAS (As-Built) of the machine and continues to use it internally. This makes the AAS an "As-IS" AAS that is the responsibility of the OPERATOR to maintain. If vulnerabilities are reported by the INTEGRATOR to the OPERATOR, it is the OPERATOR's responsibility to evaluate them for himself in his environment and to comment on them via VEX entries.

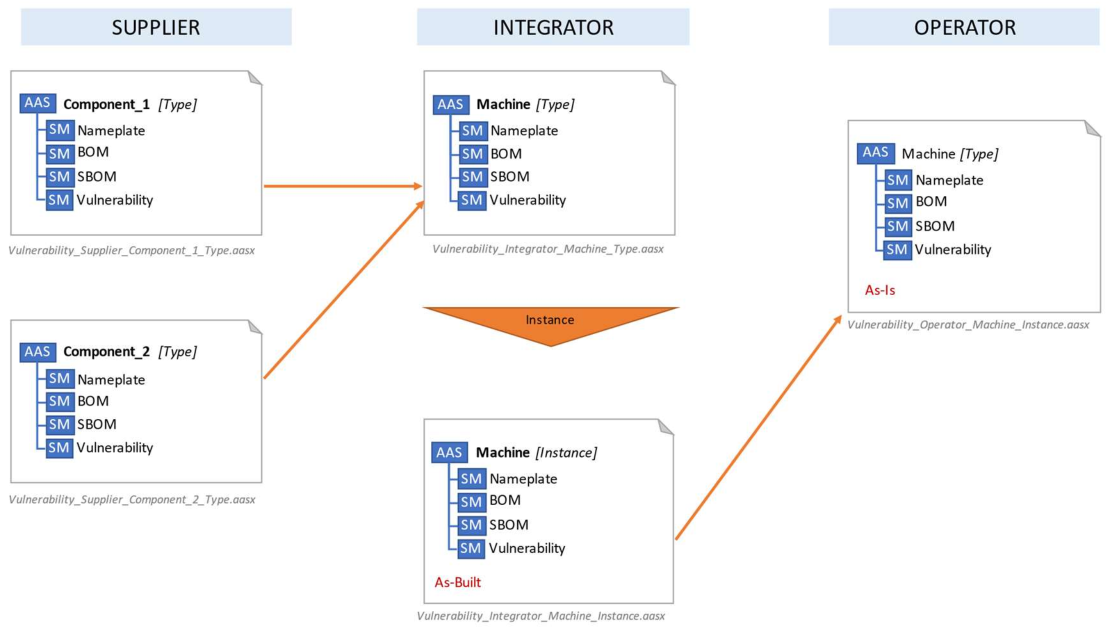

Figure 4: Big Picture description of the example

Annex A: Explanations on used table formats

General
The used tables in this document try to outline information as concise as possible. They do not convey all information on Submodels and SubmodelElements. For this purpose, the definitive definitions are given by a separate file in form of an AASX file of the Submodel template and its elements.

Tables on Submodels and SubmodelElements
For clarity and brevity, a set of rules is used for the tables for describing Submodels and SubmodelElements.

- The tables follow in principle the same conventions as in [5].
- The table heads abbreviate 'cardinality' with 'card'.
- The tables often place two informations in different rows of the same table cell. In this case, the first information is marked out by sharp brackets [] form the second information. A special case are the semanticIds, which are marked out by the format: (type)(local)[idType]value.
- The types of SubmodelElements are abbreviated: SME

SME type Submodel	**Element type**
Prop	Property
MLP	MultiLanguageProperty
Range	Range
File	File
Blob	Blob
Ref	ReferenceElement
Rel	RelationshipElement
SMC	SubmodelElementCollection
Ent	Entity

- If an idShort ends with '{00}', this indicates a suffix of the respective length (here: 2) of decimal digits, in order to make the idShort unique. A different idShort might be choosen, as long as it is unique in the parent's context.
- The Keys of semanticId in the main section feature only idType and value, such as: [IRI]https://admin-shell.io/vdi/2770/1/0/DocumentId/Id. The attributes "type" and "local" (typically "ConceptDescription" and "(local)" or "GlobalReference" and (no-local)") need to be set accordingly; see [6].
- If a table does not contain a column with "parent" heading, all represented attributes share the same parent. This parent is denoted in the head of the table.
- Multi-language strings are represented by the text value, followed by '@'-character and the ISO 639 language code: example@de.
- The [valueType] is only given for Properties.

Bibliography

[1] "Recommendations for implementing the strategic initiative INDUSTRIE 4.0", acatech, April 2013. [Online]. Available: https://www.acatech.de/Publikation/recommendations-for-implementing-the-strategic-initiative-industrie-4-0-final-report-of-the-industrie-4-0-working-group/

[2] "Implementation Strategy Industrie 4.0: Report on the results of the Industrie 4.0 Platform"; BITKOM e.V. / VDMA e.V., /ZVEI e.V., April 2015. [Online]. Available: https://www.bitkom.org/noindex/Publikationen/2016/Sonstiges/Implementation-Strategy-Industrie-40/2016-01-Implementation-Strategy-Industrie40.pdf

[3] "The Structure of the Administration Shell: TRILATERAL PERSPECTIVES from France, Italy and Germany", March 2018, [Online]. Available: https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/hm-2018-trilaterale-coop.html

[4] "Beispiele zur Verwaltungsschale der Industrie 4.0-Komponente – Basisteil (German)"; ZVEI e.V., Whitepaper, November 2016. [Online]. Available: https://www.zvei.org/presse-medien/publikationen/beispiele-zur-verwaltungsschale-der-industrie-40-komponente-basisteil/

[5] "Verwaltungsschale in der Praxis. Wie definiere ich Teilmodelle, beispielhafte Teilmodelle und Interaktion zwischen Verwaltungsschalen (in German)", Version 1.0, April 2019, Plattform Industrie 4.0 in Kooperation mit VDE GMA Fachausschuss 7.20, Federal Ministry for Economic Affairs and Energy (BMWi), Available: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/2019-verwaltungsschale-in-der-praxis.html

[6] "Details of the Asset Administration Shell; Part 1 - The exchange of information between partners in the value chain of Industrie 4.0 (Version 3.0RC01)", November 2020, [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Details-of-the-Asset-Administration-Shell-Part1.html

[7] "Semantic interoperability: challenges in the digital transformation age"; IEC, International Electronical Commission; 2019. [Online]. Available:https://basecamp.iec.ch/download/iec-white-paper-semantic-nteroperability-challenges-in-the-digital-transformation-age-en/