Contract No 608224

Deliverable 2.3
Tools for Smart Grid Cyber Security

AIT Austrian Institute of Technology • Fraunhofer AISEC • The Queen’s University Belfast
Energieinstitut an der Johannes Kepler Universität Linz • EMC Information Systems International Ltd
KTH Royal Institute of Technology • Landis + Gyr • United Technologies Research Centre
SWW Wunsiedel GmbH
<table>
<thead>
<tr>
<th><strong>Document Control Information</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title</strong></td>
</tr>
<tr>
<td><strong>Editor</strong></td>
</tr>
<tr>
<td><strong>Author(s)</strong></td>
</tr>
<tr>
<td><strong>Requested deadline</strong></td>
</tr>
</tbody>
</table>
Abstract

In this deliverable, we report the tool-chain developed to implement some of the core steps of the SPARKS risk management process for smart grids [Pro15a]. The tool-chain supports important stages of the risk management process such as: “Context Establishment”, which characterizes the use case across all layers of the smart grid model; “Threat Identification” and “Likelihood Assessment”, which identify threats and analyze their likelihood; “Consequence Identification” and “Impact Assessment”, which assess the threats consequences and their severity level.

In particular, Smart Grid Architecture Model (SGAM) extensions and interfaces have been developed to allow relevant information to be exported from the SGAM framework to other tools used in the risk management process, including the SPARKS cybersecurity simulation environment reported in [Pro16]. Additionally, an ontology-based tool has been implemented to automatically generate attack graphs, using information extracted from the SGAM framework. Finally, results of theoretic analysis of simplified use cases have been implemented in a numerical computation tool for obtaining preliminary impact assessment results, which may be used to narrow down the relevant attack patterns for detailed analysis through simulation environments.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>List of Figures</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Acronyms</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>7</td>
</tr>
<tr>
<td><strong>2 SGAM Framework extension</strong></td>
<td>11</td>
</tr>
<tr>
<td>2.1 SGAM use case instantiation</td>
<td>12</td>
</tr>
<tr>
<td>2.2 SGAM export interfaces</td>
<td>15</td>
</tr>
<tr>
<td><strong>3 Threat identification using Ontologies and Attack Graphs</strong></td>
<td>21</td>
</tr>
<tr>
<td>3.1 Modeling Examples</td>
<td>21</td>
</tr>
<tr>
<td>3.2 Threat Analysis</td>
<td>28</td>
</tr>
<tr>
<td>3.3 Vulnerability Analysis</td>
<td>33</td>
</tr>
<tr>
<td>3.4 Likelihood Assessment</td>
<td>34</td>
</tr>
<tr>
<td>3.5 Tool usage</td>
<td>35</td>
</tr>
<tr>
<td><strong>4 System Analysis: Consequence identification and preliminary impact assessment</strong></td>
<td>36</td>
</tr>
<tr>
<td>4.1 Modelling the distribution grid</td>
<td>37</td>
</tr>
<tr>
<td>4.2 Nominal operation</td>
<td>40</td>
</tr>
<tr>
<td>4.3 Impact analysis of adversarial actions</td>
<td>42</td>
</tr>
<tr>
<td><strong>5 Conclusion</strong></td>
<td>48</td>
</tr>
<tr>
<td><strong>A System Ontology</strong></td>
<td>51</td>
</tr>
<tr>
<td>A.1 Description language and conventions</td>
<td>51</td>
</tr>
<tr>
<td>A.2 System Ontology Description</td>
<td>52</td>
</tr>
<tr>
<td><strong>B Preliminary Concepts for System Analysis</strong></td>
<td>61</td>
</tr>
</tbody>
</table>
List of Figures

1. SPARKS risk management process [Pro15a] .......................................................... 8
2. The Smart Grid Architecture Model (SGAM) framework ........................................ 9
3. SGAM extension and export to other steps of SPARKS risk assessment process ........ 9
4. Building blocks of cybersecurity co-simulation environment .................................... 10
5. SGAM ICT Network Diagram for VVO use case .................................................. 12
6. VVO Use Case instances overview ................................................................. 13
7. Model translation of the ‘DSO control network’ ICT network diagram from the VVO use case into the OMNET++ network configuration .................................................. 18
8. Setup of a short system modelled using the presented SGAM Ontology .................. 22
9. Information flow and functions of the second ontology application example ............ 26
10. Root pattern for generating the attack graph ..................................................... 30
11. The relationship between the different aspects of an impact assessment ................. 36
12. Building blocks of cybersecurity co-simulation environment .................................. 37
13. A low voltage distribution grid comprised of interconnected microgrids with inverter-based DERs ................................................................. 37
14. The inverter-based DERs of a MG are controlled by a droop controller. The physical quantities are measured by sensors at each node, which then transmit their measurements (denoted by the superscript $s$) to the droop controllers. The control signal is computed based on the measurements and reference signals received by the controller (denoted with the superscript $c$). ................................................................. 39
15. An inverter-based droop controller under (a1) a reference signal attack at bus $i$, where the adversary corrupts $V_{i}^{c}$, and (a2) a measurement routing attack at bus $i$, where the adversary redirects the voltage measurement from bus $j$ to the controller at bus $i$, as if it were a measurement from bus $i$. ................................................................. 43
16. Trajectories of the voltage magnitudes under a reference signal attack at bus 3 ........ 45
17. Trajectories of the voltage magnitudes under a voltage measurement routing attack that feeds a measurement from bus 4 to bus 1, followed by a reference change at bus 3 ........ 46
18. Root-locus of the system (18) with respect to $K_{1} \geq 0$ ........................................ 47
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPE</td>
<td>Common Product Enumeration.</td>
</tr>
<tr>
<td>DER</td>
<td>Distributed Energy Resource.</td>
</tr>
<tr>
<td>HMI</td>
<td>Human Machine Interface.</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language.</td>
</tr>
<tr>
<td>PLC</td>
<td>Programmable Logic Controller.</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework.</td>
</tr>
<tr>
<td>SGAM</td>
<td>Smart Grid Architecture Model.</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier.</td>
</tr>
<tr>
<td>VVO</td>
<td>Voltage and VAR Optimization.</td>
</tr>
</tbody>
</table>
1 Introduction

In the smart grid paradigm, physical power grids become more tightly coupled to Information and Communication Technology (ICT) infrastructures, as to support new energy services and increase the flexibility of supply and demand for improved efficiency. However, the increasing use of ICT systems also introduces new cybersecurity threats and vulnerabilities, and the use of these systems to support energy services increases the impact of cyber-attacks. In this context, it is important to understand the cybersecurity risks to the smart grid, so that well-founded security requirements can be identified and controls can be implemented. The implementation of risk assessment and management practices for the smart grid is particularly challenging, given the multiple domains and players it involves: physical grid, the ICT infrastructure, the monitoring and control algorithms, the consumers, and the adversaries.

As an earlier report of the SPARKS project, a risk management process for smart grids was developed \cite{Pro15a}, which is depicted in Figure 1. The proposed process is based on the ISO/IEC 27005 information security risk management standard \cite{ISO11}, but it contains several smart grid-specific components that tackle the challenges described in the previous paragraph. Some of the key contributions of the SPARKS risk management process are present in stages such as “Context Establishment” and “Risk Assessment”. This report describes the tool-chain developed to implement some of the core steps of the SPARKS risk management process. These tools are envisioned to be used to support a risk assessment exercise on one of SPARKS’ testbeds in the forthcoming deliverable D2.6.

In particular, \cite{Pro15a} proposes the use of the SGAM framework for “Context Establishment”, i.e., for characterizing the use case across all layers of the smart grid model, including the relevant stakeholders, ICT infrastructure, and smart grid-related operations. The SGAM framework is a way of describing the architecture and functionality of the smart grid. Using a layered structure, it supports the requirement of interoperability, combining organizational, informational and technical aspects of the smart grid – three aspects that are important for risk management. SGAM provides a common understanding for different stakeholders and supports the analysis of the smart grid across several viewpoints and levels of detail. As depicted in Figure 2, the SGAM framework models the smart grid in three dimensions: There are five interoperability layers in the model: business, function, information, communication and component layer. Each layer, also called a smart grid plane, is further divided into a two dimensional grid of domains and zones. Domains reflect the physical viewpoint of the electrical delivery process, whereas zones correspond to the hierarchy of the management of this process. SGAM defines five domains (generation, transmission, distribution, Distributed Energy Resource (DER) and customer premise) and six zones (process, field, station, operation, enterprise and market).

Therefore, SGAM is a natural framework to use for “Context Establishment”, which serves as the input to the “Risk Assessment” stage that analyzes threats in terms of its likelihood (through the steps “Threat Identification” and “Likelihood Assessment”) and the respective consequences and their severity level (through the “Consequence Identification” and “Impact Assessment” steps). To support the use of SGAM for “Context Establishment” and integration with other tools, several extensions and interfaces are developed in this work, which are summarized in Figure 3. For instance, details of the communication architecture can be exported to simulation tools, e.g. OMNET++, and integrated in detailed co-simulation environments such as the one developed in SPARKS Deliverable D2.4 \cite{Pro16}. In addition to SGAM extensions and interfaces, this report also describes tools supporting “Threat Identification” and “Likelihood Assessment” as well as “Consequence Identification” and “Impact Assessment”.

The identification and analysis of threats is performed through the use of complex attack graphs, which capture the possible sequences of actions taken by an adversary to compromise components and devices. However, generating attack graphs for a real system may be a tedious and cumbersome task. To support the identification and analysis of threats, a tool has been developed to automatically generate attack graphs based on ontology reasoning. As depicted in Figure 3, the ontology-based tool for generating attack graphs uses information extracted from the SGAM framework.
Figure 1: SPARKS risk management process [Pro15a].
As discussed in [Pro15a], the identification of consequences and the assessment of their impact can be achieved with different levels of granularity and complexity, depending on the target stakeholders and the available models. For instance, a thorough assessment of the adversary’s impact can be ascertained by using the co-simulation environment described in [Pro16]. As depicted in Figure 4, the co-simulation environment requires detailed models from several domains: the power grid, the communication data network, the consumer profiles, and the adversary behavior.

However, given the complexity of the models that need to be derived, instantiated, and simulated, as well as the several different attack patterns that an adversary could exhibit, simulation-based impact assessment is quite computationally demanding and, thus, often time-consuming or infeasible. Therefore, in order to decrease the effort and time allocated to simulation-based exercises, it is desirable to begin with coarser levels of granularity to identify possible high-impact threats and attack patterns (cf. Figure 4), which can then be studied in more detail through co-simulation approaches. As discussed in [Pro15a],
Figure 4: Building blocks of cybersecurity co-simulation environment.

Theoretical system analysis is a suitable candidate for a coarser impact assessment of consequences affecting the physical grid. This report describes the theoretical analysis established for a particular use case with different attack scenarios, together with the corresponding implementation of the results for numerical computation of the impact metrics.

The outline of this report is as follows. Section 2 describes the extensions and interfaces developed for “Context Establishment” through the SGAM framework, while the ontology-based tools for automatic generation of attack graphs is reported in Section 3. The system analysis developed for a coarses impact assessment is detailed in Section 4 and final remarks are given in Section 5.
2 SGAM Framework extension

The Smart Grid Architecture Model (SGAM) is a well-established method for defining the Smart Grid use cases and their architectural realization in technology and solution unspecific manner. SGAM specifies the architecture across five interoperability layers: business, function, information, communication and component layer. The layered structure of the model supports description of the system at different abstraction levels, encapsulating both high-level business and functional context of smart grid services as well as their realization in terms of components, communication and exchanged information.

Within the SPARKS risk assessment process, SGAM Framework is used in the context establishment step, providing scope and description of the use case. We have demonstrated the process of use case modelling using the SGAM Toolbox [NET+14] by the example of Voltage and VAR Optimization (VVO) use case [Pro15a]. Model-based architecture description has a number of advantages: it provides clear and consistent system description and transparency of relations between objects/components, the model is easy to navigate and it enables traceability across all layers, e.g. the choice of components could be justified by the motivation of stakeholders. Nevertheless, the modelling is time consuming and requires great amount of manual work, especially while dealing with complex systems. Based on our experiences, SGAM so far has only been used for visual representation of the architecture, thus the effort required for modelling does not yet translate to measurable results.

Furthermore, the SGAM Framework does not clearly state how detailed the model should be in terms of components, protocols and standards definition. SGAM Toolbox methodology assumes that the outcome of SGAM is intended to be platform independent model. However, the risk assessment is limited when we provide only a generic model of the architecture. The more information we have about the system, the more detailed threat analysis and impact assessment can be performed.

In order to address the above mentioned shortcomings, we propose to extend the context establishment step in two ways:

1. extending the SGAM model with use case instantiation,
2. exporting the model to other tools in risk assessment process.

Figure 3 presents how the process of risk assessment could be supported by instantiation of the SGAM use case and export of the model to other tools. Assuming that the generic use cases for smart grids are already available in common use case repository (already as SGAM models), they can be instantiated for particular system (one of interest for the stakeholder executing risk assessment). For instance, a DSO, e.g. SWW, is interested in introducing new service (use case) to the grid. To this end, a use case is instantiated on the infrastructure operated by the DSO, providing details about the technological solutions for communication, software and hardware components. Having the instantiated model of the use case, it can be exported to tools supporting other steps of risk assessment process, specifically:

- identification of threats and threat analysis,
- identification of consequences and impact assessment.

In addition, we propose an export to Verinice, an Information Security Management System (ISMS) that supports 27005 risk assessment. It could be used either as an alternative or a complementary tool to the SPARKS risk assessment.

In summary, we propose two ways of supporting the SPARKS risk assessment process. Firstly, we improved the context establishment by use case instantiation, the more concrete system description. Secondly, by exporting model to other tools, we extend the use of SGAM beyond only visual representation of the system.

Deliverable 2.3 Page 11 of 62
2.1 SGAM use case instantiation

In this section we present the method for instantiating the SGAM use case on existing or foreseen system. We do not propose a completely new tool for SGAM instantiation. Instead, we suggest extending the SGAM toolbox by additional diagrams and attributes for SGAM elements. We benefit from the flexibility of UML and since SGAM toolbox is a plugin for Enterprise Architect, we make use of other UML functionalities that come with Enterprise Architect. Extensions were dictated by the other tools, to the exported information is aligned with the next steps of the risk assessment process.

The first extension, already provided in the SGAM toolbox, is the network topology diagram. SGAM framework considers communication between components only by direct connection, without details about the underlying network infrastructure. However, in some cases communication of more than one pair of components is realized by the same network and it would be useful to reflect this fact in the model.

Secondly, we use the tagged values to enrich the description of components with additional attributes such as domain, zone, type, version, operating system (for computers). It is also possible to define new stereotypes of components for special types of devices, applications, software etc.

Finally, we propose another layer for instantiated model, with created instances of each component and schematic topology of the system. Each instance of the component can be specified further with additional attributes, e.g., each instance of the controller can have different operating system. The above-mentioned extensions are explained in more detail below using the example of the VVO use case.

---

**Figure 5: SGAM ICT Network Diagram for VVO use case**
Example: VVO use case instantiation  The VVO use case describes control loop at the Data Management System (DMS) level and can be summarized in three steps. Firstly, the measurements from the grid are collected, including voltage and reactive power measurements from Distributed Energy Resources (DERs), settings from the Substations and consumption data from smart meters. Secondly, the DMS calculates the optimized settings based on measurements from the first step and then send out the commands with new set points to Substations, DERs and flexible loads. The complete model of the VVO use case is provided in [Pro15a], in this section, though, we will focus only on the extended content.

Following the SGAM framework, in the first version of the VVO SGAM components were connected by direct ICT connection relation. In the newly revised version of the component layer, depicted in Figure 5(a), includes two ICT Network components Enterprise and DSO Control networks, that are specified in separate ICT Network diagrams in Figure 5(b) and c). Enterprise Network is used for communication between DMS and Meter Data Management Server (MDMS), whereas DSO Control Network for gathering the measurements from the grid and sending out the optimized set points.

Modelling the ICT infrastructure allows us to include the communication equipment in the list of components and their characteristics. Information about network is meaningful for risk assessment and required for interface to OMNET++ and ontology-based attack trees. In Table 2 the new communication equipment components are listed. Another novel extension is the instance layer of the VVO use case shown in Figure 6. In this layer we create instances of the components defined in SGAM component layer, e.g. one instance of DMS, two instances of PSN (primary substation nodes) etc. Additionally, we instantiate network components and topology. The attributes, communication relations (protocols) as well as infor-
information flows of certain component are derived for each instance. The attributes required for certain type of component can be defined with generic default value and further specified for each instance of this component.

Table 2: Communication Equipment for VVO use case

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
<th>Stereotype</th>
<th>Domain</th>
<th>Zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENet Gateway</td>
<td>Gateway for the Enterprise Network</td>
<td>ICT Gateway</td>
<td>Distribution</td>
<td>Enterprise</td>
</tr>
<tr>
<td>DSONet Gateway</td>
<td>Gateway for the DSO control network</td>
<td>ICT Gateway</td>
<td>Distribution</td>
<td>Enterprise</td>
</tr>
<tr>
<td>PSNet Gateway</td>
<td>Gateway for the primary substations</td>
<td>ICT Gateway</td>
<td>Distribution</td>
<td>Station</td>
</tr>
<tr>
<td>FL Gateway</td>
<td>Gateway for the flexible loads</td>
<td>ICT Gateway</td>
<td>Customer Premises</td>
<td>Field</td>
</tr>
<tr>
<td>ENet switch</td>
<td>Switch for the Enet</td>
<td>Switch</td>
<td>Distribution</td>
<td>Enterprise</td>
</tr>
<tr>
<td>PSNet switch</td>
<td>Switch for the Primary substation network</td>
<td>Switch</td>
<td>Distribution</td>
<td>Station</td>
</tr>
<tr>
<td>DSONet switch</td>
<td>Switch for the DSO control network</td>
<td>Switch</td>
<td>Customer Premises</td>
<td>Field</td>
</tr>
</tbody>
</table>

Table 3 lists all the instances in VVO example with additional attributes for each type of component. Apart from the SGAM specific domain and zone attributes, we can specify operating system and its version for each computer and device, brand and firmware for switches, firewall type, version of software etc.

Table 3: VVO use case instances with additional attributes

<table>
<thead>
<tr>
<th>Label</th>
<th>Component</th>
<th>Stereotype</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMS_1</td>
<td>DMS, Distribution Management System</td>
<td>Computer</td>
<td>Domain: Distribution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zone: Enterprise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Operating System: Windows</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OS version: WindowsServer2012</td>
</tr>
<tr>
<td>DMS_App_1</td>
<td>DMS Control Algorithm</td>
<td>Software</td>
<td>Domain: Distribution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zone: Enterprise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>version: 3.1</td>
</tr>
<tr>
<td>MDMS</td>
<td>MDMS, Meter Data Management System</td>
<td>Computer</td>
<td>Domain: Distribution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zone: Enterprise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Operating System: Linux</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OS version: Ubuntu Server 15.04</td>
</tr>
<tr>
<td>PSN_1</td>
<td>PSN, Primary Substation System Node</td>
<td>Computer</td>
<td>Domain: Distribution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zone: Station</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Operating System: Windows</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OS version: WindowsServer2012R</td>
</tr>
<tr>
<td>PSN_2</td>
<td>PSN, Primary Substation System Node</td>
<td>Computer</td>
<td>Domain: Distribution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zone: Station</td>
</tr>
<tr>
<td>Model/Device</td>
<td>Domain/Zone</td>
<td>Operating System</td>
<td>OS version</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td>DER_Wind_1</td>
<td>Station</td>
<td>Windows Server 2012</td>
<td></td>
</tr>
<tr>
<td>DER_Wind_2</td>
<td>Field</td>
<td>Alpine Linux 3.3.1</td>
<td></td>
</tr>
<tr>
<td>MV_FL_1_C</td>
<td>Customer Premises</td>
<td>Linux</td>
<td></td>
</tr>
<tr>
<td>FL_G_1</td>
<td>Field</td>
<td>Debian 8.3</td>
<td></td>
</tr>
<tr>
<td>DSONet_switch_1</td>
<td>Distribution</td>
<td>Linux</td>
<td></td>
</tr>
<tr>
<td>DSO_Net_GW_1</td>
<td>Enterprise</td>
<td>Linux</td>
<td>Ubuntu</td>
</tr>
<tr>
<td>PSNet_G_1</td>
<td>Station</td>
<td>Windows RRAS</td>
<td></td>
</tr>
<tr>
<td>PSNet_S_1</td>
<td>Station</td>
<td>Cisco ASA</td>
<td>5510</td>
</tr>
<tr>
<td>ENet_G_1</td>
<td>Enterprise</td>
<td>Linux</td>
<td></td>
</tr>
<tr>
<td>FW_1</td>
<td>Enterprise</td>
<td>Cisco ASA</td>
<td>1.1.2.0</td>
</tr>
</tbody>
</table>

### 2.2 SGAM export interfaces

Model-driven development using the SGAM toolbox necessitates the transformation of models to specific tools which are dedicated to facilitate analysis involved at different stages of the risk assessment.
process. In order to enable re-use of the information and models developed in the SGAM toolbox, trans-
formations scripts are proposed as shown in the Figure [3]. Model transformations are at the heart of
model-driven approach that assists the rapid adaptation, evolution and reuse of models at various levels
of detail. As the SGAM interoperability layers represent the various use case perspectives, model trans-
formations allow to extract information relevant in format that is executable in other tools. In this section
we describe the transformation of the SGAM models into the format readable to three different tools:

- RDF Ontology for threat analysis
- OMNET++ for co-simulation based impact assessment
- Verinice for Information System Security Management evaluation

Following subsections describe the parsers for transforming the SGAM models into the RDF ontology,
OMNET++, Verinice and illustrate the transformation by the VVO use case.

2.2.1 Interface to RDF Ontology for threat analysis

The ontology-based tool for threat identification and analysis in SPARKS risk assessment process re-
quires input in RDF Turtle format. In the RDF data is specified in form of subject-predicate-object
triples. The description of the SPARKS ontology definition for SGAM is given in A.2. It specifies how
SGAM entities, such as components, connections, protocols, information objects, are expressed in terms
of RDF triples.

The SGAM to ontology transformation script is written in python and provided as sgamlib.py module.
The procedure of exporting SGAM use case into ontology can be summarized in following steps:

1. Firstly, the SGAM use case needs to be instantiated, since the essential information required for
ontologies is the instances of components.

2. The Enterprise Architect supports export of the UML models into XMI format (XML for UML).
Even though we are interested in the instances, still all SGAM layers are exported in order to
retrieve necessary information about the use-case-specific components, relations and information
flows.

3. The exported XML file is then an input to sgamlib, which parses the model and imports the selected
information to internal representation of SGAM content. We have defined three classes that XML
content is mapped to: components, connections and information objects.

4. Following that, the export into RDF format is handled using RDFlib python module, which stores
triples in the graph.

5. Components, connections (protocols) and information objects are translated into RDF triples using
A.2 and added to the graph.

6. Finally, the RDF graph is serialized and exported in a turtle format into a given output file.

Example: Exporting instantiated VVO use case into RDF turtle file

The VVO use case has been instantiated[3] and exported to the xml file (vvomodel.xml). Firstly, we parse
XML for all Node elements and import them as components. The component class that we proposed
in sgamlib consists of following attributes: label, name, description, stereotype, OS, OS version, zone,
domain. By the example of DMS_1 instance of DMS component have attributes: DMS_1, DMS, Distrib-
component attributes are then translated into RDF triple in the following way:
ex:DMS_1 a sgam:Computer, owl:NamedIndividual ;
rdfs:label "DMS" ;
sgam:inDomain sgam:DistributionDomain ;
sgam:inZone sgam:EnterpriseZone ;
rdfs:comment "Distribution Management System" .

Next, we search for all Association elements to be mapped into connection class with attributes: type, source, target, protocol. For example, the connection from DMS (source) to MDMS (target) over the protocol IEC 61968 is mapped to RDF as:

ex:IEC_61968_DMS_MDMS a sgam:Protocol, owl:NamedIndividual ;
rdfs:label "IEC_61968 Protocol between DMS and MDMS" .

Finally, we retrieve information objects from all associations of type Information Flow and their attributes: name, description, source, target, data model (protocol). For example, information flow between MDMS (source) and DMS (target) convey smart meter consumption measurements (name) information object over IEC 61968 (data model). Information Object is translated into three objects in SGAM ontology:

ex:smcm_DMS a sgam:LocalInformationObject, owl:NamedIndividual ;
rdfs:label "smart meter consumption measurements" ;
sgam:isStoredAt ex:DMS_1 ;
rdfs:comment "smart meter consumption measurements stored at DMS" .

ex:smcm_MDMS a sgam:LocalInformationObject, owl:NamedIndividual ;
rdfs:label "smart meter consumption measurements" ;
sgam:isStoredAt ex:MDMS_1 ;
rdfs:comment "smart meter consumption measurements stored at MDMS" .

ex:smcm_MDMS_DMS a sgam:SharedInformationObject, owl:NamedIndividual ;
rdfs:label "smart meter consumption measurements" ;
sgam:exchangedVia ex:IEC_61968_DMS_MDMS ;
sgam:hasReader ex:smcm_DMS ;
sgam:hasWriter ex:smcm_MDMS ;
rdfs:comment "smart meter consumption measurements transferred from MDMS to DMS" .

2.2.2 Interface to OMNET++

OMNET++ [VH08] is a simulation framework and development environment widely used in the research community for building communication network simulations. The OMNET++ framework components are programmed in C++ with ability to be assembled into a larger modules/models and packeted into specialized libraries for simulation of wired and wireless communication networks. The INET library is essential OMNET module that contains various Internet stack models, for example IPv4/IPv6 network stack, transport layer protocols (TCP, UDP, SCTP), wired and wireless layer protocols (Ethernet, PPP,
Moreover, within the SPARKS project, the OMNET tool has been integrated into the joint OMNET-GridLabD co-simulation framework for detailed impact assessment of cyber attacks on the Smart Grid infrastructure (the details of the co-simulation framework are given in D2.4). The impact assessment is an important sub-process of the overall SPARKS Risk Management process. The context of the risk assessment process is established using the SGAM toolbox in the deliverable D2.2. In order to automatically translate the use case information already available in the SGAM toolbox into the co-simulation framework, the following steps are defined:

1. The component layer of the SGAM architecture needs to be enhanced with the ICT Network Topology Diagrams describing the communication networks deployed between different components;
2. The component layer has be exported as an XML file of UML 1.x format, supported by the Enterprise Architect tool;
3. The parser tool has to be executed on the exported XML file, in order to convert it into the OMNET readable format, called the Network Description (NED) file.

In the example below, the SGAM component model of the VVO use case, extended with the ICT network diagram is converted in the NED file and shown in the OMNET framework.

**Example: Exporting the ICT network diagram to the NED file** In this example, the SGAM component model of the VVO use case, extended with the ICT network diagram, is converted in the NED file and shown in the OMNET framework. In Table 1, we first show how the parser maps individual components of ICT network diagram into the network components of the OMNET/INET framework, while Figure 2 shows visual representation of the original and translated model.

![Figure 7: Model translation of the 'DSO control network' ICT network diagram from the VVO use case into the OMNET++ network configuration](image)

### 2.2.3 Interface to Verinice

Verinice [Gmb] is an open-source tool for managing information security. Verinice can be used for:

- Establishing, maintaining and improving an Information Security Management System (ISMS) based on ISO 27001
Table 4: Mapping of ICT network diagram elements onto the OMNET/INET network elements

<table>
<thead>
<tr>
<th>ICT network diagram elements</th>
<th>OMNET/INET network elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch Node</td>
<td>EtherSwitch model: is a model of an Ethernet switch with CSMA/CA support</td>
</tr>
<tr>
<td>Router Node and ICT Gateway Node</td>
<td>Router model: is a IPv4 router that supports wireless, Ethernet and Point-to-Point interfaces</td>
</tr>
<tr>
<td>Component Layer Node, Workstation Node and Server Node</td>
<td>StandardHost model: is a model of any end-node component running transport layer protocols (TCP, UDP) and any applications on top of them</td>
</tr>
<tr>
<td>Network Cloud Node</td>
<td>InternetCloud: is a router that can delay or drop packets (while retaining their order) based on which interface the packet enters and exist the router</td>
</tr>
<tr>
<td>Ethernet Connection</td>
<td>Ether100M: a model of an Ethernet link with 100 Mbit/s of physical layer data rate</td>
</tr>
<tr>
<td>Network Connection and Serial Connection</td>
<td>DatarateChannel: a model of general network connection link with the configurable parameter for propagation delay, data rate speed, bit error rate and packet error rate</td>
</tr>
</tbody>
</table>

- performing risk analysis based on ISO 27005
- auditing, document management and report generation

Verinice allows the users to perform full risk analysis of the information assets and derive further actions based on the results. The tool allows adding of threats and various vulnerabilities from existing databases. Verinice supports risk identification according to ISO 27005 standard, which can be used for further risk analysis. Moreover, the Verinice tool gives possibility to maintain a register of information assets and processes. In order to facilitate the SPARKS risk assessment process for Smart Grids, an export interface was developed that allows a user to export already identified information assets during the use case description process, from the SGAM toolbox into the Verinice. These elements can be exported into the comma-separated values (CSV) file readable for Verinice tool in the following steps:

1. The component layer of the SGAM architecture should be exported as an XML file of UML 1.x format, supported by the Enterprise Architect tool;
2. The script should be executed on the exported XML file, in order to convert it into the CSV file;
3. The CSV file should imported into the existing Verinice project.

**Example: Exporting the VVO use case Assets to Verinice** In order to enable the automatic import of assets from use case modelled in SGAM to Verinice, we defined SGAM elements corresponding to the types of assets available in verinice, i.e. information, software, and physical assets. In terms of
physical and software assets we export SGAM components, or, if available, their instances, whereas
SGAM Information Objects are of type information asset. The example of VVO use case assets in CSV
format for Verinice is listed below:

```
ext-id;name;type;description
1;DMS_1;Physical;Distribution Management System
2;DMS_App_1;Software;DMS Control Algorithm
3;PSN_1;Physical;Primary Substation Node 1
4;MDMS;Physical;Meter Data Management System
5;smcm;Information;Smart meter consumption measurements
```
3 Threat identification using Ontologies and Attack Graphs

In the SPARKS risk assessment process, threat identification using attack trees takes a prominent role. In SPARKS deliverable D2.2 [Pro15a] a set of patterns that facilitate the process of a systematic construction of attack trees in presented. The patterns are derived from the SGAM system description and mainly capture the structural aspects of attacks. They have to be complemented by attack trees for technical attacks on a very local basis (e.g. the specifics of an exploit), coming from expert knowledge and preferably reused in form of tree databases.

While these patterns facilitate the generation of complex attack graphs, applying them to a real system is still a tedious task. Therefore, tools supporting this step of generating these trees are required. The systematic nature of this generation process supports automation with tools.

Machine-based reasoning based on ontologies received a lot of attention in the last years and is a subject of study on its own. There are plenty of approaches to use this technique also in the context of information security, as already presented by the work on semantic threat graphs in SPARKS deliverable D2.2 [Pro15a].

In this chapter, a method for automatic attack tree generation based on ontology reasoning is presented. A precise ontology for modelling the system under consideration in terms of SGAM is given. As any ontology, it is extensible if additional aspects need to be modelled. The ontology not only captures the functional aspects of the smart grid, but also allows modelling the technical implementation aspects if available. As the latter is crucial for identifying attack vectors from a technical perspective, adding this information to the model is beneficial for the outcome of the threat analysis.

Based on the VVO use-case, which was already used as a running example in D2.5 [Pro15b] and D2.2 [Pro15a], a model is established using the developed ontology (cf. Section 3.1.2). In Section 3.2 the ontology description in combination of queries to the ontology to generate the attack tree patterns. This process is also illustrated by applying the methods to the VVO use-case example. In addition, the derived system description is linked to a vulnerability database to emphasise the benefits of the presented approach. With this step it is possible to identify unpatched vulnerabilities in the system, also.

In this document, further methods for (1) evaluating the leaves and (2) propagating the values through the graph are not discussed in detail. For the first aspect, various methods exist, which will be reviewed briefly. For the second aspect, the solution is straightforward.

The ontologies, examples, and queries are implemented using RDFLib, SPARQLWrapper and FuXi libraries. Scripts written in Python 2.7 are used to realise the interconnections between the various applications. The open-source ontology editor Protégé is used to model and maintain the structure of our ontology and HermiT as a semantic reasoner and for consistency checks. The knowledge graph is stored on an Apache Jena Fuseki Server, hosting a persistent triple store.

In the following sections, the ontology described in Appendix A is illustrated for two use-cases, after which the ontology is used to automatically identify threats and analyse vulnerabilities, and to support the likelihood assessment.

3.1 Modeling Examples

In this section, two use-cases are modelled and presented as examples. The first example is a short, detailed system description (cf. Section 3.1.1), while the second example in Section 3.1.2 reflects the abstract VVO Use-Case, referenced at D2.2, Threat and Risk Assessment Methodology [Pro15a].
3.1.1 Short System Example

The following example is a model of the system depicted in Figure 8. It includes two computers, a switch, a Programmable Logic Controller (PLC), a network description as well as running some sample software and a patch.

![Diagram of the short system example](image)

**Example 1. Model of a short example system**

```
ex:ETH_1 rdf:type sgam:SwitchedWiredCommunicationNetwork ,
owl:NamedIndividual ;
rdfs:label "LAN1" ;
rdfs:comment "The local Ethernet at the example Backend" ;
sgam:switchedBy ex:Switch_1 .

ex:Switch_1 rdf:type sgam:CommunicationSwitch ,
owl:NamedIndividual ;
rdfs:comment "The switch of network ETH_1" ;
rdfs:label "SWITCH 1" ;
sgam:inZone sgam:FieldZone;
sgam:inDomain sgam:DistributionDomain .

ex:PLC_1 rdf:type sgam:ProgrammableLogicDevice ,
owl:NamedIndividual ;
rdfs:label "IO Control PLC" ;
rdfs:comment "Sample PLC including localization and connection" ;
sgam:inZone sgam:FieldZone;
sgam:inDomain sgam:DistributionDomain;
sgam:connectedTo ex:ETH_1 .

ex:OPC_Client_1 rdf:type sgam:Workstation ,
owl:NamedIndividual ;
rdfs:label "OPC Client 1" ;
rdfs:comment "Example Workstation" ;
sgam:inZone sgam:FieldZone;
sgam:inDomain sgam:DistributionDomain;
sgam:connectedTo ex:ETH_1 .
```
ex:Windows_1 rdf:type sgam:OperatingSystem,
    owl:NamedIndividual;
    rdfs:comment "The windows instance running on OPC_Client_1,
        including version and patch information";
    rdfs:label "Windows 7 SP1";
    sgam:isVersion "6.1.7601";
    sgam:patchedBy ex:Patch_1;
    sgam:runsOn ex:OPC_Client_1 .

ex:AS_Modbus_OPC_Client_1 rdf:type sgam:ApplicationSW,
    owl:NamedIndividual;
    rdfs:comment "Application instance, running on OPC_Client_1";
    rdfs:label "Modbus/TCP OPC Client";
    sgam:isVersion "1.42";
    sgam:runsOn ex:OPC_Client_1 .

ex:Patch_1 rdf:type sgam:SWPatch,
    owl:NamedIndividual;
    rdfs:comment "A patch including an uri";
    rdfs:label "KB2872339";
    rdfs:isDefinedBy

ex:OPC_Server_1 rdf:type sgam:Server,
    owl:NamedIndividual;
    rdfs:label "OPC Server 1";
    rdfs:comment "Server including localization and
        network connection information";
    sgam:connectedTo ex:ETH_1;
    sgam:inZone sgam:FieldZone;
    sgam:inDomain sgam:DistributionDomain .

ex:Windows_2 rdf:type sgam:OperatingSystem,
    owl:NamedIndividual;
    rdfs:comment "Windows instance running on the Server";
    rdfs:label "Windows Server 2012 R2";
    sgam:runsOn ex:OPC_Server_1 .

ex:AS_Modbus_OPC_Server_1 rdf:type sgam:ApplicationSW,
    owl:NamedIndividual;
    rdfs:comment "Modbus/TCP OPC Server";
    rdfs:label "OPC Server Software 1";
    sgam:runsOn ex:OPC_Server_1 .

ex:TCPIP_1 rdf:type sgam:Protocol, owl:NamedIndividual;
    rdfs:label "TCP/IP Connection 1";
    rdfs:comment "The TCP/IP protocol run at ETH1";
    sgam:usesCommunicationNetwork ex:ETH_1 .
3.1.2 VVO Use Case Example

This example describes the VVO Use-Case and is split into three segments; the component and communication layer, the information layer, and the function layer. The modelled information and function flow is depicted in Figure 9.

Example 2. VVO - Component and communication layer (see D2.2 Chap. 4.1)

```
ex:SCS_Actor_1 rdf:type sgam:Actor, owl:NamedIndividual; rdfs:label "Substation Control Centre (SCS)";
sgam:inDomain sgam:DistributionDomain;
sgam:inZone sgam:EnterpriseZone.

ex:DER_Actor_1 rdf:type sgam:Actor, owl:NamedIndividual; rdfs:label "Distributed Energy Resources (DER)";
sgam:inDomain sgam:DERDomain;
sgam:inZone sgam:FieldZone.

ex:CC_Actor_1 rdf:type sgam:Actor, owl:NamedIndividual; rdfs:label "Control Centre (CC)";
sgam:inDomain sgam:DistributionDomain;
sgam:inZone sgam:EnterpriseZone.

ex:PE_Actor_1 rdf:type sgam:Actor, owl:NamedIndividual; rdfs:label "Power Equipment (PE)";
sgam:inDomain sgam:DistributionDomain;
sgam:inZone sgam:StationZone.

ex:FL_Actor_1 rdf:type sgam:Actor, owl:NamedIndividual; rdfs:label "Flexible Load (FL)";
sgam:inDomain sgam:CustomerDomain;
sgam:inZone sgam:ProcessZone.

ex:DMS_1 rdf:type owl:NamedIndividual, sgam:Server; rdfs:label "Distribution Management System 1";
sgam:inDomain sgam:DistributionDomain;
sgam:inZone sgam:EnterpriseZone;
sgam:connectedTo ex:ETH_Net_1.

ex:MDMS_1 rdf:type sgam:Server, owl:NamedIndividual; rdfs:label "Meter Data Management System 1";
sgam:inDomain sgam:DistributionDomain;
sgam:inZone sgam:EnterpriseZone;
sgam:connectedTo ex:ETH_Net_1.

ex:ER_1 rdf:type sgam:CommunicationRouter, owl:NamedIndividual; rdfs:label "Enterprise Network Router";
sgam:inDomain sgam:DistributionDomain;
sgam:inZone sgam:EnterpriseZone;
```
ex:ETH_Net_1 rdf:type sgam:WiredCommunicationNetwork,
    owl:NamedIndividual;
    rdfs:label "Enterprise Domain Network";
    rdfs:comment "The enterprise level devices,
                   namely MDMS and DMS can be connected
                   with an LAN and a (COTS) router that also
                   provides a connection to the Internet.".

ex:MODBUS_Net_1 rdf:type sgam:WiredCommunicationNetwork,
    owl:NamedIndividual;
    rdfs:label "Modbus line between Enterprise and PSN.";
    rdfs:comment "The PSN has a dedicated Modbus line to the ER.".

ex:GSM_Net_1 rdf:type sgam:WirelessCommunicationNetwork,
    owl:NamedIndividual;
    rdfs:label "GSM Network between PSN and MV FL";
    rdfs:comment "The PSN has a GSM module,
                   to communicate with the MV FL.".

ex:GSM_Net_2 rdf:type sgam:WirelessCommunicationNetwork,
    owl:NamedIndividual;
    rdfs:label "GSM Network between PSN and DER Wind SG GW";
    rdfs:comment "The PSN has a GSM module,
                   to communicate with the DER Wind SG GW.".

ex:PSN_1 rdf:type sgam:Server, owl:NamedIndividual;
    rdfs:label "Primary Substation Node 1";
    sgam:inDomain sgam:DistributionDomain;
    sgam:inZone sgam:StationZone;
    sgam:connectedTo ex:MODBUS_Net_1, ex:GSM_Net_1,
    ex:GSM_Net_2.

ex:MV_FL_1 rdf:type sgam:Generator, owl:NamedIndividual;
    rdfs:label "Medium Voltage Flexible Load";
    sgam:inDomain sgam:CustomerDomain;
    sgam:inZone sgam:ProcessZone;
    sgam:connectedTo ex:GSM_Net_1;
    sgam:operatedByActor ex:FL_Actor_1.

ex:DER_Wind_SG_GW_1 rdf:type sgam:Generator, owl:NamedIndividual;
    rdfs:label "DER Wind Smart Grid Gateway";
    sgam:inDomain sgam:DistributionDomain;
    sgam:inZone sgam:StationZone;
    sgam:connectedTo ex:GSM_Net_2;
    sgam:operatedByActor ex:DER_Actor_1.
Example 3. VVO - Information layer example (see D2.2 Fig. 5)

Figure 9: Information flow and functions of the second ontology application example.

```rml
ex:Local_Info_1 rdf:type sgam:LocalInformationObject , owl:NamedIndividual ;
    rdfs:comment "Smart Meter Consumption Measurements (SM CM) at the MDMS" ;
    rdfs:label "SM CM @ MDMS_1" ;
    sgam:isStoredAt ex:MDMS_1 .

ex:Local_Info_2 rdf:type sgam:LocalInformationObject , owl:NamedIndividual ;
    rdfs:comment "Distributed Voltage Measurement (DVN) at the PSN" ;
    rdfs:label "DVN @ PSN_1" ;
    sgam:isStoredAt ex:PSN_1 .

ex:Local_Info_3 rdf:type sgam:LocalInformationObject , owl:NamedIndividual ;
    rdfs:comment "Smart Meter Consumption Measurements (SM CM) at the DMS" ;
    rdfs:label "SM CM @ DMS_1" ;
    sgam:isStoredAt ex:DMS_1 .
```
ex:Local_Info_4 rdf:type sgam:LocalInformationObject ,
owl:NamedIndividual ;
rdfs:comment "Distributed Voltage Measurement (DVM)
at the DMS" ;
rdfs:label "DVN @ DMS_1" ;
sgam:isStoredAt ex:DMS_1 .

ex:Local_Info_5 rdf:type sgam:LocalInformationObject ,
owl:NamedIndividual ;
rdfs:comment "calculated tap setting command (TSC)" ;
rdfs:label "TSC @ DMS_1" ;
sgam:isStoredAt ex:DMS_1 .

ex:Local_Info_6 rdf:type sgam:LocalInformationObject ,
owl:NamedIndividual ;
rdfs:comment "The TSC at the PSN" ;
rdfs:label "TSC @ PSN_1" ;
sgam:isStoredAt ex:PSN_1 .

ex:Shared_Info_1 rdf:type sgam:SharedInformationObject ,
owl:NamedIndividual ;
rdfs:label "SharedInformationObject 1" ;
sgam:hasReader ex:Local_Info_3 ;
sgam:hasWriter ex:Local_Info_1 ;
sgam:exchangedVia ex:Webservice_Con_1 ;
rdfs:comment "The SM CM is transferred from Local_Info_1 (MDMS) via Webservice_Con_1 to Local_Info_3 (DMS)" .

ex:Shared_Info_2 rdf:type sgam:SharedInformationObject ,
owl:NamedIndividual ;
rdfs:label "SharedInformationObject 2" ;
sgam:hasReader ex:Local_Info_4 ;
sgam:hasWriter ex:Local_Info_2 ;
sgam:exchangedVia ex:IEC61850_Con_1 ;
rdfs:comment "The DVM is transferred from Local_Info_2 (PSN_1) via IEC61850_Con_1 to Local_Info_4 (at DMS_1)" .

ex:Shared_Info_3 rdf:type sgam:SharedInformationObject ,
owl:NamedIndividual ;
rdfs:label "SharedInformationObject 3" ;
sgam:hasReader ex:Local_Info_6 ;
sgam:hasWriter ex:Local_Info_5 ;
sgam:exchangedVia ex:IEC61850_con_1 ;
rdfs:comment "The calculated TSC is transferred from Local_Info_5 (where it was calculated) via IEC61850_con_1 to Local_Info_6" .

ex:Local_Function_1 rdf:type sgam:LocalFunction ,
owl:NamedIndividual ;
rdfs:label "TSC calculation function" ;
Example 4. VVO - Function Layer example (see D2.2 Fig. 5)

ex:Function_1 rdf:type sgam:Function, owl:NamedIndividual;
  rdfs:label "Collect Data";
  sgam:dependsOnInformation ex:Local_Info_1, ex:Local_Info_2;
  rdfs:comment "This function depends on the SM CM at the MDMS and the DVM at the PSN".

ex:Function_2 rdf:type sgam:Function, owl:NamedIndividual;
  rdfs:label "Calculate Setpoints";
  sgam:dependsOnInformation ex:Local_Info_3, ex:Local_Info_4, ex:Local_Info_5;
  rdfs:comment "This function depends on the available Input and Output at the DMS".

ex:Function_3 rdf:type owl:NamedIndividual, sgam:Function;
  rdfs:label "Send Setpoints";
  sgam:dependsOnInformation ex:Local_Info_6;
  rdfs:comment "This function depends on the right TSC at the PSN".

3.2 Threat Analysis

As described in SPARKS deliverable D2.2 [Pro15a] Chapter 5.2.2, SPARQL queries are used to synthesize attack trees from the ontology-based knowledge base. The outcome of this process is an attack graph that visualizes the potential targeted components and network connections an attacker has to impair in order to violate an asset.

The tool was developed to support the attack tree generation. The output is generated as a CSV file, which could be then processed further to a graphical representation. To demonstrate the proposed analysis method, the information of the presented Short System example (cf. Section 3.1.1) and the VVO Use-Case example (cf. Section 3.1.2) are used.

3.2.1 Root Asset Identification

The first step of the presented threat analysis is the identification of the roots of the trees to guide the analysis. These roots are called root assets. As described in SPARKS deliverable D2.2 [Pro15a], every information that leaves the scope of the analysis, meaning the boundaries of the information system under examination, should be considered as a root asset:

1. information being transferred to other (sub-)networks,
2. control values that are output to the physical process by an actuator, and
3. values that are communicated to humans, e.g., via a Human Machine Interface (HMI).
The identification of root assets is considered in the context of single functions (which somehow correspond to use cases). A function \( f \) is passed as input to the tool, and a list of potential root assets is generated. With this modelling, it is only possible to identify information sinks, but not if this information is actually an output to an external system. Therefore, the list of root assets needs still some filtering by an expert.

A root asset has the following characteristics in the presented model:

- it is a LocalInformationObject, which is a local representation (reader) of a SharedInformationObject
- the function \( f \) depends on the SharedInformationObject
- it is a sink in the graph spanned by local functions, readers, and writers of SharedInformationObjects (see Figure 9 for an example of this graph in the VVO use case).

This is captured by the following Query 1:

```sql
SELECT DISTINCT ?rootAsset ?label
WHERE {
  %f% sgam:dependsOnInformation ?sharedObject.
  ?sharedObject sgam:hasReader ?rootAsset.
  ?rootAsset rdf:type sgam:LocalInformationObject .
  filter not exists {?noSharedObject sgam:hasWriter ?rootAsset} .
  filter not exists {?noLocalFunction sgam:hasInput ?rootAsset}.
  ?rootAsset rdfs:label ?label
}
```

Listing 1: Query for Output Information.

**Example 5.** In our VVO use case example, when using `ex:Function_2` as input, the output of this query would be:

`ex:Local_Info_6 "TSC @ PSN_1"`

which is the output of the TSC at the PSM, i.e., the information of interest. The root node can then be named:

A. Manipulation of "TSC @ PSN_1"

### 3.2.2 Attack Tree Synthesis

After identification of the root assets, the attack graph is synthesised by creating individual attack trees for every root asset. As described in SPARKS deliverable D2.2 [Pro15a], patterns are used to create the attack trees. The steps describing the process of constructing the queries to generate these patterns automatically are detailed next. As an illustration, the steps are applied to the root pattern depicted in Figure 10.

The child nodes of the attack goal node are therefore:
A: Attack Goal

A.1 Consider manipulation of the output emitting component
A.2 Consider manipulation of the output at the component determining it.
A.3 Consider manipulation of components where the information is routed through
A.4 Consider manipulation of communication links that are involved in the information transfer

Figure 10: Root pattern for generating the attack graph.

- A single node for the manipulation at the component that is responsible for the output of the root asset.
- Nodes for the manipulation at components that hold a value that determines the value of the root asset.
- Nodes for the manipulation at components where this information is routed through.
- Nodes for the manipulation at communication that are involved in this transfer.

Each group of nodes is generated by a separate query, the results are then combined to generate the graph.

Node “Consider manipulation of the component emitting the output”

At first, the component that emits the output is identified. The query for this is straightforward, identifying the component where the asset is stored.

```
Query2(%asset%) = [ 
  SELECT ?component ?label 
  WHERE { 
    %asset% sgam:isStoredAt ?component . 
  } 
]
```

Listing 2: Query for nodes of type “A.1” (cf. Figure 10).

Example 6. Using ex:Local_Info _6 as input in the VVO use case example, the output of this query is be as expected:

```
ex:PS_1 "Primary Substation Node 1"
```

This result can be added as a child to the root node:

A.1 Manipulation of Primary Substation Node 1

Nodes of type “Consider manipulation of the output at the component determining it” For this set of branches, the set of components that could be compromised in order to manipulate the computation of the asset under consideration needs to be determined. This comprises two steps. In the first step, all LocalInformationObjects that have an influence on the LocalInformationObject (asset) that is given as an input are computed. This set of information objects is computed using two queries, in a recursive manner. Dependency is either because a local function generates the value, or because of the connection through a SharedInformationObject.
Query3(%info%) =
  SELECT ?input
  WHERE {
    ?localFunction sgam:hasOutput %info% .
    ?localFunction sgam:hasInput ?input .
  }]

Query4(%info%) =
  SELECT ?writer
  WHERE {
    ?shared_object sgam:hasWriter ?writer .
    ?shared_object sgam:hasReader %SharedInformationObject%
  }

Function1(%info%) =
  all_infos := [ %info% ]
  do
    for every new i in all_infos:
      add result of Query3(i) to all_infos
      add result of Query4(i) to all_infos
    until there is no new infos added to the list
  return all_infos

Listing 3: Query for the determining component.

Next, for every identified information object, the component holding the value is determined:

Query5(%info%) =
  SELECT ?component
  WHERE {
    %LocalInformationObject% sgam:isStoredAt ?component
  }

Function2(all_infos) =
  components = empty set
  for every i in all_infos:
    add Query5(i) to components
  return components

Listing 4: Query for the component storing the LocalInformationObject.

Example 7. For the VVO use case, using ex:Local_Info_6 as input to Function1, a list of LocalInformationObjects is the output:

ex:Local_Info_5
ex:Local_Info_4
ex:Local_Info_3
ex:Local_Info_1
ex:Local_Info_1

Issuing Function2 on this list, yields to following components as a result:
Sorting out duplicates, a child node for `ex:DMS_1` and `ex:MDMS_1` can be added to the tree.

**Nodes of type “Consider manipulation of components where the information is routed through”**

In a similar fashion, nodes for the manipulation of those components, where the information is routed through, can be added. The routing information is stored in the protocol stacks of the SGAM description.

```sql
Query6(%asset%) = [
  SELECT ?component
  WHERE {
    ?shared_object sgam:hasReader %asset% .
  }
]
```

Listing 5: Query for the component storing the LocalInformationObject.

Query 6 yields a list of components that involved in the information transfer of the information asset. As in Example 11, the nodes can be added to the attack tree. The query as to be executed for every information object that was identified in the last step (“determining component”).

**Nodes of type “Consider manipulation of the communication links that are involved in that information transfer”**

At this branch, the communication links that are involved in the information transfer are considered. This includes all potential forms of eavesdropping or tampering on wired and wireless connections and buses between the previously identified components. This query must also be repeated for all returned LocalInformationObjects, uncovering the corresponding SharedInformationObjects and their used networks.

```sql
Query7(%asset%) = [
  SELECT ?network
  WHERE {
    ?shared_object sgam:hasReader %asset% .
  }
]
```

Listing 6: Query for the component storing the LocalInformationObject.

### 3.2.3 Applying Standard Patterns and Expert Knowledge

Having the attack patterns for the root asset, the library patterns from SPARKS deliverable D2.2 [Pro15a] for compromising components and connections can be directly and recursively added to the attack tree.

In addition, the generated tree requires review and refinement by an expert. Depending on the type of components, networks, or protocols, specific subtrees containing this expert knowledge can be added. However, the tedious and error-prone construction of the tree core is automatically generated from the system description.
3.3 Vulnerability Analysis

A simple vulnerability analysis is performed by linking the objects at the leaves and nodes of the attack tree with knowledge about their individual vulnerabilities. The analysis shall support the evaluation of the leaf nodes of the tree in the likelihood analysis (see Section 4).

3.3.1 Link to Vulnerability Databases

To link structured data from different sources, it is necessary to use equivalence statements, subset relationships or common Uniform Resource Identifier (URI). Therefore, matching statements are defined to enable import into the established knowledge graph.

Because there is no Smart Grid specific vulnerability database yet, conventional IT and network vulnerability databases are used until a more specific database is available, which then can be integrated.

An example of linking is the following enhancement of two operating system classes to match with the identifier from the Common Product Enumeration (CPE) ID, which is used for the NVD CVEs.

Example 8. Matching SGAM ontology classes with the CPE


After creating this matching table, a command line tool is developed to import the NVD CVE XML feed (https://nvd.nist.gov/download.cfm#CVE_FEED) into the established knowledge graph.

3.3.2 Assessing Vulnerabilities and Patch Level

The developed tool surveys each attack tree, using the knowledge graphs in order to create a list of all possible vulnerabilities, that are not known to be mitigated, for each element of the tree. As the available information about the system as well as vulnerabilities are both typically incomplete or coarse grained, a worst case assumption is made from here, leading to a pessimistic perspective wherever a secure state could not be assumed from the available knowledge. This is the case when, for example, only the operating system family of a system is known, leading to the assumption that the system is threatened by all vulnerabilities that affect at least one member of this family. The aim of this process is to evaluate all possible paths that might lead to an impairment of the identified assets, and initiate the creation of a semantic threat graph for each item.

A part of the threat identification process is the search for missing patches that is performed by the following query, for which no example is given, as the two examples are too superficial to be modelled, and contain too little patch information, resulting in a list of nearly all possible patches from the database:

```sql
1 SELECT ?computer ?os ?patch
2 WHERE {
3   ?computertype rdfs:subClassOf* sgam:Computer.
4   ?OStype rdfs:subClassOf* sgam:OperatingSystem.
5   ?dev rdf:type ?computertype.
6   ?os rdf:type ?OStype.
7   ?patch rdf:type sgam:SWPatch.
```
filter not exists {?os sgam:patchedBy ?patch} .
}

Listing 7: Query for the missing operating system patches of a computer.

An example for finding potential vulnerabilities of a specific component is given for the AS Modbus OPC Server Software from the Short System Example presented at Section 3.1.1.

Query for the vulnerabilities of a specific software  In Listing 8 an example query is presented which finds vulnerabilities of a specific software in the knowledge graph.

```sql
SELECT ?cveVul ?cveDescription
WHERE {
  ex:AS_Modbus_OPC_Server_1 rdf:type ?itemtype .
  ?cveVul cve:hasDescription ?cveDescription
}
```

Listing 8: Query for the vulnerabilities of a specific software.

Example 9. For the OPC Server Software (ex:AS_Modbus_OPC_Server_1) from the Short System Example the vulnerability CVE-2010-4709 is identified as unpatched.

Whenever a complete, vulnerable path to a root asset is found, it is up to the operator to block this way (e.g. by complementing the system (respectively the model) with countermeasures like patches, network separation, etc.

3.4 Likelihood Assessment

The next step in the SPARKS risk assessment framework is the likelihood assessment of the leaf nodes of the tree, and the propagation of the results to the root. While the presented approach gives new methods for threat and vulnerability analysis, for the likelihood assessment we refer to existing methods like TVRA or HMG IS1 to rate the value of the assets and the criticality of leafs and nodes. Reasons therefore are, that the information from the available databases is to coarse grained to allow qualified statements which transitions (between functions, components, software and the network) are truly possible by one or a combination of the assigned vulnerabilities. Further, expert input (clearly benefiting from the provided vulnerability and attack path information) is necessary to consider control flows, including the storage of credentials that enable access to other elements without usage of a further vulnerability, the impact and consequences, as described in Section 4. An aggregation and conservation of such vulnerability correlations, control flows and scaled impacts inside a new database poses an interesting but challenging future research topic.

Using the outcome of the expert feedback for each identified step of the attack path, an algorithm should be implemented to propagate and correlate the determined likelihood ratings through the remaining graph in order to reveal the critical attack vectors. By questioning the knowledge graph about the specific vulnerabilities of each element, an individual Semantic Threat Graph, (that should be enriched by countermeasures like patches, policies etc) could be created.
3.5 Tool usage

In order to evaluate a system using the tool-set developed in this deliverable, the analysis system requires a fully working installation of Python 2.7 as well as the RDFLib, SPARQLWrapper and FuXi libraries. The open-source ontology editor Protégé is used to model and maintain the structure of the developed ontology and HermiT as a semantic reasoner and for consistency checks. The knowledge graph is stored at an Apache Jena Fuseki Server, hosting a persistent triple store.

In order to assess a system using this tool chain, the following steps have to be performed:

1. Create a model of the system under evaluation using the SGAM ontology.
2. Run the attack tree synthesis tool to identify the potential attack vectors.
3. Download and import the NVD CVE Vulnerability XML Feed.
4. Run the vulnerability assignment tool to link the graph elements with their potential vulnerabilities.
5. Evaluate the output considering the impact and control flows.
4 System Analysis: Consequence identification and preliminary impact assessment

The impact assessment stage of the risk assessment methodology reported in [Pro15a] is comprised of different aspects, as depicted in Figure 11. In particular, two key aspects are the consequences of threat scenarios and the metrics to assess their severity.

For instance, for the VVO use case, an important category is the “Quality of Supply”, where relevant consequences relate to variations of the voltage magnitude due to an attack, which may be measured in terms of voltage stability, peak voltage value, or percentage voltage variation. These consequences and metrics will be revisited in this section, for a specific use case of voltage control under adversarial actions. In particular, the techniques reported in this section can be used to preliminarily assess the impact of different attack scenarios on a voltage control scheme, through the use of numerical tools that do not require extensive simulations.

The preliminary impact assessment methodology is as follows: firstly, the distribution grid and the voltage control scheme are modelled, which will provide the basis for the impact assessment. To establish a baseline performance, the nominal closed-loop system (i.e., without any attack) is analyzed and characterized in terms of the relevant consequence metrics: voltage stability and peak voltage values, for instance. Secondly, the system behaviour under each attack scenario is modelled; these models are then used to analyze and characterize the performance of the closed-loop system under attack with respect to the attacked devices, the grid and controller parameters, and the impact metrics. As a result, the analysis can be directly applied to systems with different parameters, and preliminary impact metrics may be computed without resorting to computationally intensive simulations.

Integration with Smart Grid Cyber Security Simulation Environment

As discussed in Section 6.2.5 of D2.2 [Pro15a], the preliminary results obtained from a system analysis may be used to reduce the complexity of more detailed analysis. More specifically, once relevant attack scenarios have been highlighted by system analysis, a detailed impact assessment of these scenarios may be performed by using fine-grained cyber-security simulation environments.

For instance, recalling the building blocks of the SPARKS cyber-security simulation environment reported in D2.4 [Pro16], which are shown in Figure 12 the relevant attack scenarios identified by a preliminary system analysis can serve as an input to narrow down the attack patterns analyzed in the simulation environment. In particular, for a given attack scenario, a system analysis could characterize which compromised device results in the most severe impact, as well as how that device has been used by the adversary (e.g., by adding a constant bias, or another signal shape, or even shutting down the device).
Figure 12: Building blocks of cybersecurity co-simulation environment.

Figure 13: A low voltage distribution grid comprised of interconnected microgrids with inverter-based DERs.

### 4.1 Modelling the distribution grid

The power distribution system is considered to be a set of interconnected MGs that may be connected to the main grid through the feeder substation, where each MG may contain several inverter-based distributed energy resources (DER) and loads. The distribution system is depicted in Figure 13, where each MG is represented by a bus and the multiple DERs and loads within a given MG are lumped together and modeled as a single DER and load, respectively.
4.1.1 Power grid model

Although Figure 13 depicts a line network, we consider generic connected topologies where the network is characterized by the undirected graph $G(V, E)$, where $V$ is the vertex set, $E$ is the edge set, and $N_i = \{ j \in V : (i, j) \in E \}$ denotes the neighbor set of the $i$-th bus. In this system, the states are defined as $V_i$ and $\theta_i$, which are voltage magnitude and voltage angle of the $i$-th bus, respectively, and $i \in V$.

**Assumption 1.** In the power distribution network under study, the following assumptions are made:

1. The three-phase power network is balanced (so that it can be represented as an equivalent single-phase system);
2. All $N$ buses are assumed to be inverter buses [SCRG14], each represented by $V_i$ and $\theta_i$ for $i = 1, \ldots, N$.

Under Assumption 1, the active and reactive power injections at bus $i$ is given respectively by

\begin{align}
P_i &= V_i^2 G_i - \sum_{j \in N_i} V_i V_j (G_{ij} \cos(\theta_{ij}) + B_{ij} \sin(\theta_{ij})), \\
Q_i &= -V_i^2 B_i - \sum_{j \in N_i} V_i V_j (G_{ij} \sin(\theta_{ij}) - B_{ij} \cos(\theta_{ij})),
\end{align}

in which $G_{ij} = R_{ij} / (R_{ij}^2 + X_{ij}^2) \geq 0$ and $B_{ij} = -X_{ij} / (R_{ij}^2 + X_{ij}^2) \leq 0$ are, respectively, the conductance and susceptance of the transmission line between the $i$-th and $j$-th buses, and $R_{ij}$ and $X_{ij}$ are resistance and reactance of the same line between the same buses, respectively. In addition, self-conductance and self-susceptance are defined as $G_i = G_{ii} + \sum_{j \in N_i} G_{ij}$ and $B_i = B_{ii} + \sum_{j \in N_i} B_{ij}$, respectively. Note that the angle difference between node $i$ and $j$, $\theta_i - \theta_j$, is simply written as $\theta_{ij}$ in the rest of the section.

**Assumption 2.** In the power distribution system under study, the transmission line impedances are assumed to have the same ratio $R_{ij} / X_{ij} = -G_{ij} / B_{ij} = \rho \geq 0$ for all lines $(i, j) \in E$.

The line ratio is related to the nature of the power system: power systems with inductive transmission lines ($R_{ij} << X_{ij}$) have a small ratio $\rho > 0$, while systems with resistive lines have a higher ratio. The latter is often the case for medium- and low-voltage distribution grids. Since the line ratio $\rho$ depends on the transmission line characteristics, Assumption 2 naturally holds for systems with homogeneous transmission lines that have similar characteristics. Moreover, Assumption 2 is commonly used in the literature, often restricted to the case of purely inductive lines ($\rho = 0$) [SPDB13b, SOA+14].

4.1.2 PV inverters and control architecture

In terms of the voltage and phase-angle dynamics, each MG $i$ is modeled as a pair of single integrators

\begin{align}
\tau_i \dot{V}_i(t) &= u_{V_i}(t), \\
\tau_\theta \dot{\theta}_i(t) &= u_{\theta_i}(t),
\end{align}

where $\tau_i > 0$ and $\tau_\theta > 0$ are the inverter’s time-constants and $u_{V_i}(t)$ and $u_{\theta_i}(t)$ are the control signals computed by the droop controller at time $t \geq 0$. The architecture of the control system is illustrated in Figure 14, which depicts the measurements and reference signals available to each controller. Using the capabilities of the local inverter-based DERs, each MG is controlled by a droop controller, which receives the reference signal computed remotely ($V_i^*$ as the reference voltage for the $i$-th bus) and measurements ($V_j$ and $\theta_j$, as the voltage magnitude and voltage angle of the $j$-th bus, respectively) through the communication network, using a suitable communication protocol such as the IEC 61850.
Figure 14: The inverter-based DERs of a MG are controlled by a droop controller. The physical quantities are measured by sensors at each node, which then transmit their measurements (denoted by the superscript $s$) to the droop controllers. The control signal is computed based on the measurements and reference signals received by the controller (denoted with the superscript $c$).

Since we are mainly interested in the voltage dynamics of the power system, the phase-angle dynamics are neglected and the following assumptions is considered throughout the remainder of this work.

**Assumption 3.** The phase-angle differences between any neighboring nodes, $\theta_{ij}$ for $(i, j) \in E$, is assumed to be constant.

Note that the analysis under Assumption [3] may be interpreted as a local analysis in scenarios where the phase-angles remain in the neighborhood of the original equilibrium point. Additionally, we highlight that the assumption is valid if there exists a time-scale separation between the phase-angle and the voltage dynamics.

To compute the voltage control signals, we consider the voltage quadratic droop controller [SPDB13a, equation (7)] described by

$$u_{V_i}(t) = -\kappa_i V_i^c(t) (V_i^c(t) - V_i^{c*}(t)) - Q_i^c(t),$$

(3)

where $\kappa_i > 0$ is the droop control gain and $V_i^c(t)$, $Q_i^c(t)$, and $V_i^{c*}(t)$ are the voltage measurement, reactive injection measurement, and voltage reference signal with respect to bus $i$, respectively, that are received by the droop controller, as illustrated in Figure [14]. Under nominal operation, these signals match the corresponding physical variables and reference signals, i.e., $V_i^c(t) = V_i(t)$, $Q_i^c(t) = Q_i(t)$,
and \( V_i^* (t) = V_i^* (t) \) (\( V_i^* (t) \) is sent by a higher level controller which is called Secondary controller). Under nominal operation, the closed-loop dynamics of the \( i \)-th MG are given by the differential equations

\[
\tau_i \dot{V}_i = -\kappa_i V_i (V_i - V_i^*) - Q_i
\]

\[
= -V_i \left( \kappa_i V_i - \kappa_i V_i^* + \sum_{j \in V} l_{ij}(\theta) V_j \right), \quad \forall i = 1, \ldots, N,
\]

where the time argument has been omitted. In addition, under the Assumption 2, the parameter \( l_{ij} \) is written as

\[
l_{ij} = \begin{cases} B_{ij}(\rho \sin(\theta_{ij}) + \cos(\theta_{ij})), & i \neq j \\ -B_i, & i = j. \end{cases}
\]

Denoting \( V = [V_1 \ldots V_N]^\top \), \( \tau = [\tau_1 \ldots \tau_N]^\top \), \( \kappa = [\kappa_1 \ldots \kappa_N]^\top \), and \( [V] \) as the diagonal matrix with \( V_i \) as the \( i \)-th diagonal entry, the voltage dynamics under the quadratic droop control can be written in vector form as

\[
[\tau] \dot{V} = [V] ([\kappa] V^* - ([\kappa] + L(\theta)) V),
\]

where the matrix \( L(\theta) \) is defined as \( [L(\theta)]_{ij} = l_{ij}(\theta) \).

4.2 Nominal operation

In this section, we provide necessary and sufficient conditions on the power system parameters so that the linearized dynamics hold the desirable properties discussed in Appendix B: positivity and row-diagonally dominance. These properties are then used to establish the asymptotic stability of the linearized system. Moreover, they play an important role when studying the power system under the attack scenarios in subsequent sections.

4.2.1 System properties

First we derive necessary and sufficient conditions for the linearized system (7) to be positive, which requires the following assumption.

**Assumption 4.** The maximum phase difference between any two neighboring nodes, defined as

\[
\Delta_{\theta} := \max_{(i,j) \in E} |\theta_{ij}|,
\]

satisfies the inequality \( \Delta_{\theta} < \pi/2 \).

Recall that the constraint \( \Delta_{\theta} < \pi/2 \) is an operational requirement for any conventional power system [SOA+14], which is required for the stability of the phase-angle dynamics. Under the previous assumptions, the following result is established.
Theorem 1. Consider the power distribution network under study, having active and reactive power injections (1) at bus \(i\) with \(\Delta \theta < \pi/2\), and applying the quadratic droop controller (4) for each MG. Then a necessary and sufficient condition for the corresponding linearized system (7) to be positive is

\[
\rho \leq |\cot(\Delta \theta)|.
\]  

(9)

Proof. This proof and the subsequent ones are omitted. \(\square\)

Note that several of the properties of positive systems stated in Appendix B have important consequences in the context of power systems and, in particular, the voltage dynamics. Letting the input \(u(t)\) be the voltage reference at one individual bus and \(x(t)\) and the voltages of all buses, the closed-loop system being positive implies that an increase in the voltage reference \(u(t)\) translates to an increase in all the voltages \(x(t)\). Hence, there is no contradictory effect where a desired increase in voltage at one bus inadvertently decreases the voltage in other buses. Additionally, positivity of the closed-loop system also reduces the voltage overshoots in response to step changes in the voltage reference.

Remark 1. While the latter discussion motivates positivity as a desirable system feature, Theorem 1 provides a design objective for the phase-angle controller that ensures positivity of the voltage dynamics, namely the inequality (9). In fact, since the line ratio \(\rho\) is a system parameter that depends solely on the transmission lines’ characteristics, (9) can be interpreted as a bound on the phase-angle differences that is parameterized by the line ratio \(\rho\). Rewriting (9) as \(|\tan(\Delta \theta)| \leq \rho^{-1}\), we have that a resistive system with a large \(\rho\) yields a strict bound on the maximum phase-angle difference \(\Delta \theta\), while a purely inductive system with \(\rho = 0\) does not constrain the phase-angle difference.

Next we characterize necessary and sufficient conditions for a linearized positive system to be row-diagonally dominant.

Lemma 1. Suppose the linearized system (7) is positive. The system (7) is row-diagonally dominant if, and only if, the following inequality holds

\[
\kappa_i + |B_{ii}| \geq (\sqrt{\rho^2 + 1} - 1) \sum_{j \in N_i} |B_{ij}|.
\]  

(10)

These properties play important roles in the characterization of the attack impacts, and they are also used in analyzing the stability of the linearized system.

4.2.2 Stability of the power system

Next we establish the stability of the linearized system, using the positivity and row-diagonally dominance properties of the linearized system. Specifically, when the system is positive, the next result states the necessary and sufficient conditions for stability and then shows that row-diagonally dominance ensures stability.

Theorem 2. Consider the linearized dynamics of the power system (7) and suppose the system is positive. Then the following statements hold:

1. the system is asymptotically stable if and only if there exist positive scalars \(\xi_i > 0\) such that the following inequality holds for all \(i = 1, \ldots, n:\)

\[
\xi_i| - \kappa_i + B_i| > \sum_{j \in N_i} \xi_j| - B_{ij}(\rho \sin(\theta_{ij}) + \cos(\theta_{ij}))|;
\]
2. The system is asymptotically stable if it is row-diagonally dominant, i.e., the following inequality holds for all \( i = 1, \ldots, n \):
\[
|\kappa_i + B_i| > \sum_{j \in N_i} |B_{ij}(\rho \sin(\theta_{ij}) + \cos(\theta_{ij}))|.
\]

**Remark 2.** Note that we may not have control on self-susceptance \( (B_{ii}) \) and it belongs to the interval \([0, \tilde{B}_{ii}]\), so to be more conservative, the sufficient condition in Proposition 2 can be written as:
\[
\kappa_i \geq \sum_{j \in N_i} (\sqrt{\rho^2 + 1} - 1)|B_{ij}|.
\] (11)

**Remark 3.** It could be interesting to characterize conditions on (7) under which \( V \) satisfies \(|V - 1| < \delta\). This problem is related to the validity of (7), which assumes that \( V \) is positive. It also relates to how \( V^{c*} \) should be constrained so that the system is safe.

### 4.3 Impact analysis of adversarial actions

Recently, [KMM+15] investigated the implementation of cyber attacks against a common application-level protocol in Smart Grid applications, the IEC 61850. In the considered attack scenario, cyber adversaries built a custom tool to execute man-in-the-middle attacks and manipulate data transmitted to a photovoltaic power inverter, thus affecting the physical power system. In this subsection, the potential consequences of such cyber attacks on the power grid are investigated. As stated in the beginning of the section, this analysis may serve as an input to more detailed cyber security co-simulation environments, to characterize and narrow down relevant attack patterns to be analyzed in further detail.

The following subsections follow a similar structure and each one considers a specific attack scenario. In particular, each subsection begins by describing the adversarial model and how it affects the droop controller. Then, the impact of the attack is characterized based on properties of the linearized system, such as stability and input-output induced-norm. Such characterizations also aim at identifying which sets of attacked nodes yield possibly higher impacts, thus indicating which threats may pose a high risk to the system. The theoretical analysis is then complemented with numerical simulations of the attack scenarios in the nonlinear system (6).

#### 4.3.1 Voltage reference attack

The present scenario considers an adversary that injects false-data into the communication network supporting the control system. In particular, we consider reference signals attacks defined as follows.

**Definition 1 (Reference signal attack).** In a reference signal attack on bus \( j \), the reference signal of bus \( j \) is corrupted, as depicted in Figure [15], so that
\[
V_{j}^{c*}(t) = u^a(t),
\] (12)

where the signal \( u^a(t) \) is defined by the adversary. Furthermore, the control signal at bus \( j \) under a reference signal attack is given by
\[
u_{V_j} = -\kappa_j V_j^{c*} (V_j^{c*} - u^a(t)) - Q_j^c.
\] (13)

The impact of the attack is measured in terms of the resulting changes to the voltage magnitude at another bus \( i \neq j \) in the network, i.e. \( V_i \). The resulting linearized system can be expressed as
\[
\dot{x}(t) = Ax(t) + \tau_j^{-1} \kappa_j e_j u^a(t)
\]
\[
y_i(t) = e_i^T x(t)
\] (14)
Figure 15: An inverter-based droop controller under (a1) a reference signal attack at bus $i$, where the adversary corrupts $V^c_\star_i$, and (a2) a measurement routing attack at bus $i$, where the adversary redirects the voltage measurement from bus $j$ to the controller at bus $i$, as if it were a measurement from bus $i$.

where $A = -[\tau]^{-1}([\kappa] + L(\theta))$ and $e_i \in \mathbb{R}^n$ is the $i$-th column of the $n$-dimensional identity matrix. In particular, we quantify the attack’s impact as the maximum deviation of $y_i(t)$ caused by a corrupted reference $u^a(t)$ that is bounded as $|u^a(t)| \leq 1$. In fact, as discussed in Appendix B, this metric corresponds to the $L_\infty$-induced norm of (14). For power systems satisfying the conditions of Theorem 1 and Lemma 1, i.e., the system (14) is positive and stable, the following characterization of the worst-case attack naturally follows from Lemma 3.

Lemma 2. Consider the linearized power system (7), which is assumed to be positive and asymptotically stable, and suppose that bus $j$ is under a reference signal attack. Let $H_{ij}(s)$ be the transfer function of (14). The worst-case impact on bus $i$ of a reference signal attack on bus $j$, characterized as the $L_\infty$-induced norm of (14), is given by $H_{ij}(0) = -\tau_j^{-1}\kappa_j e_i^\top A^{-1} e_j = \tau_j^{-1}\kappa_j [-A^{-1}]_{i,j}$.

Such characterization of the worst-case impact can be leveraged to compare different attacks and identify scenarios with higher impact. In particular, supposing bus $j$ is attacked, we are interested in assessing which other bus $i \neq j$ is most affected by the attack. That is, we seek to compute

$$i^* = \arg \max_i H_{ij}(0) = \arg \max_i [-A^{-1}]_{i,j},$$

where the common factor $\tau_j^{-1}\kappa_j$ has been omitted.

Although solving such problem would, in general, require the computation of all entries of $-A^{-1}$, specific power system topologies admit simpler solutions. Specifically, for power systems whose topology corresponds to a line graph, the following result establishes that the $L_\infty$-induced norm $[-A^{-1}]_{i,j}$ decreases as the distance between $i$ and $j$ increases.

Theorem 3. Consider a power system whose topology corresponds to a line graph and the respective linearized dynamics (7) are positive and row-diagonally dominant. Furthermore, suppose the droop...
controller at bus $j$ is under a reference signal attack. Then the $L_\infty$-induced norm of the linearized system under attack (14) is given by $H_{ij}(0) = \tau_j^{-1} \kappa_j [-A^{-1}]_{i,j}$, which satisfies the monotonicity conditions
\begin{align*}
[-A^{-1}]_{i,j} > [-A^{-1}]_{i+1,j}, \quad \forall j \leq i \\
[-A^{-1}]_{i,j} > [-A^{-1}]_{i-1,j}, \quad \forall j \geq i.
\end{align*}

(15)

Considering line graphs, using the results of Theorem 3, we conclude that the impact of a reference attack decays as the distance to the attacked bus increases. Moreover, the bus most affected by the attack at bus $j$, defined as $i^* = \arg \max_i H_{ij}(0)$, corresponds to one of the neighboring buses of $j$, i.e., $i^* = \arg \max_{i \in \{j-1, j+1\}} [-A^{-1}]_{i,j}$.

**Numerical example**

To illustrate the impact of the attack on the reference signal, we consider an islanded 4-bus power system with a line topology, as depicted in Figure 13 with $N = 4$, and assume identical power lines, loads, and inverters. The power system is characterized by (1) with the parameters $\rho = 0.5$, $B_{ij} = -0.2$, and $G_{ij} = -\rho B_{ij}$ for all edges $(i, j) \in E$ and $B_{ii} = -0.001$ and $G_{ii} = \rho |B_{ii}|$ for all buses. The power inverters are modeled by (2) and (3) with parameters $\tau_i = 10^{-4}$, $\tau_{\theta_i} = 10^{-2}$, and $\kappa_i = 0.2$ for all buses.

To motivate Assumption 3, two sets of simulations are performed: one where Assumption 3 is satisfied, since the phase-angle differences are constant throughout the simulation of the voltage dynamics and are given by $\theta_{12} = -0.11\text{rad}$, $\theta_{23} = 0.045\text{rad}$, and $\theta_{34} = -0.11\text{rad}$; another where a suitable droop controller is used for the phase-angle dynamics, with the previous set of phase-angle differences as an initial condition, and with noise in the voltage measurements.

The voltage dynamics are described by the nonlinear differential equations (6), with the corresponding linearized dynamics characterized by (7) with
\[
A = 10^{-4} \begin{bmatrix}
-4.01 & 1.88 & 0 & 0 \\
2.1 & -6.01 & 2.04 & 0 \\
0 & 1.95 & -6.01 & 1.88 \\
0 & 0 & 2.1 & -4.01
\end{bmatrix}.
\]

Clearly, the system is positive and row-diagonally dominant. Since the diagonal entries $A$ are negative, the system is also asymptotically stable.

Now consider the reference signal attack scenario where the voltage reference transmitted to bus 3 is corrupted by an adversary, as per Definition 1. Following the discussion in this section, we seek to assess which buses, other than bus 3, are most affected by such attack. From Lemma 2, the worst-case impact of such attack on a given bus $i$ in the network corresponds to $H_{i3}(0) = -K_3 \tau_3^{-1} e_i^\top A^{-1} e_3$. In present example, the set of worst-case gains to buses 1, 2, and 4 are given by $H_{13}(0) = 0.09$, $H_{23}(0) = 0.19$, and $H_{43}(0) = 0.25$, respectively. As stated by Theorem 4 for line graphs, the largest worst-case impact takes place at one of the neighbors of bus 3, which here corresponds to bus 4.

The decrease of the impact as the distance to bus 3 increases is visible on the voltage trajectories of the nonlinear system under a reference attack on bus 3, as depicted in Figure 16a. A similar behavior is also observed with varying phase-angles and measurement noise, as illustrated in Figure 16b.

### 4.3.2 Voltage measurement routing attack

Here we consider an adversary that is able to redirect truthful data from its intended destination to another receiving bus in the network. In particular, we consider measurement routing attacks defined as follows.
Definition 2 (Measurement routing attack). In a measurement routing attacks on bus $i$, the adversary redirects the voltage measurement from bus $j$ as if it were a measurement from bus $i$, which is captured by having

$$V^c_i = V^s_j = V_j.$$  

Furthermore, the corresponding control signal under attack is described as

$$u_{V_i} = -\kappa_i V^s_j (V^s_j - V^c_i) - Q^c_i,$$
$$u_{V_k} = -\kappa_k V^c_k (V^c_k - V^c_j) - Q^c_k, \quad \forall k \neq i.$$  

(16)

The resulting linearized system under a measurement routing attack at bus $i$ can be expressed as

$$\dot{x}(t) = (A - \tau^{-1}_i \kappa_i e_i (e_j - e_i)^\top) x(t),$$  

(17)

where the term $-\tau^{-1}_i \kappa_i e_i (e_j - e_i)^\top x(t)$ can be interpreted as replacing the nominal feedback term $\tau^{-1}_i \kappa_i V_i$ by the corrupted feedback $\tau^{-1}_i \kappa_i V_j$ at bus $i$. In fact, such attack scenario can be rewritten as the following static output-feedback law

$$\begin{align*}
\dot{x}(t) &= (\bar{A}_i x(t) + \tau^{-1}_i \kappa_i e_i u(t) \\
y_j(t) &= e_j^\top x(t) \\
u(t) &= -\kappa_i y_j(t),
\end{align*}$$  

(18)

where the matrix $\bar{A}_i$ is independent of the control gain $\kappa_i$.

Note that the closed-loop system under attack (17) is no longer positive, nor diagonally dominant, since we have $[A - \tau^{-1}_i \kappa_i e_i (e_j - e_i)^\top]_{i,j} = -\kappa_i < 0$ and $[A - \tau^{-1}_i \kappa_i e_i (e_j - e_i)^\top]_{i,i} = [A]_{ii} + \kappa_i$. As such, the results of Section 4.2.2 may not be used to establish the stability of (17). In fact, the closed-loop system (17) may indeed be unstable for certain values of $\kappa_i \geq 0$, as established by the following result.

Theorem 4. Consider a power system whose linearized dynamics (7) are positive. Furthermore, suppose the droop controller at bus $i$ is under a measurement routing attack that feeds the controller with the voltage measurement from the $j$-th bus, as per Definition 2. Then there exists a control gain $\kappa_i \geq 0$ for which the linearized system under attack (17) is unstable if $\text{dist}(j,i) \geq 2$, where $\text{dist}(j,i)$ is the shortest length between buses $i$ and $j$. 

Figure 16: Trajectories of the voltage magnitudes under a reference signal attack at bus 3.
Time [s] × 10^{-3}
0 1 2 3 4 5 6
0.9
0.92
0.94
0.96
0.98
1
1.02
1.04
1.06
1.08
1.1
V \(c^\star\) 3
V1
V2
V3
V4

(a) Constant phase-angles

(b) Varying phase-angles and measurement noise

Figure 17: Trajectories of the voltage magnitudes under a voltage measurement routing attack that feeds a measurement from bus 4 to bus 1, followed by a reference change at bus 3.

Theorem 4 establishes the existence of a positive gain \(\kappa_i\) for which the attacked system becomes unstable when \(\text{dist}(j,i) \geq 2\). In other words, for a particular choice of control gains, a measurement routing attack on buses that are not adjacent can lead the system to instability and have severe consequences. Similar results were derived in [AR14] under the assumption that the open-loop system remains diagonally dominant, which does not hold for the present system.

**Numerical example**

Recall the example described in Section 4.3.1 and consider the measurement routing attack scenario where an adversary replaces the voltage measurement at bus 1 with the voltage measurement of bus 4, which is modeled by (17) with \(i = 1\) and \(j = 4\). The resulting closed-loop state matrix is

\[
\tilde{A}_1 - K_1 e_1 e_4^\top = 10^{-4} \cdot \begin{bmatrix}
-2.01 & 1.88 & 0 & -2 \\
2.1 & -6.01 & 2.04 & 0 \\
0 & 1.95 & -6.01 & 1.88 \\
0 & 0 & 2.1 & -4.01
\end{bmatrix},
\]

which is clearly not diagonally dominant, nor positive. Despite stability, the lack of such properties leads to contradictory behaviors, as illustrated by the response to a step-change in one reference, depicted in Figure 17. Despite the increase in the reference signal, bus 1 further decreased its voltage.

As stated in Theorem 4, since the distance between buses 1 and 4 is greater than 1, there exists a gain \(K_1 \geq 0\) for which the system under attack becomes unstable. This is illustrated through the corresponding root-locus depicted in Figure 18.
Figure 18: Root-locus of the system (18) with respect to $K_1 \geq 0$. 
5 Conclusion

This deliverable reported the tool-chain that has been developed to implement some of the core steps of the SPARKS risk management process for smart grids [Pro15a]. The tool-chain supports important stages of the risk management process such as: “Context Establishment”, which characterizes the use case across all layers of the smart grid model; “Threat Identification” and “Likelihood Assessment”, which identify threats and analyze their likelihood; “Consequence Identification” and “Impact Assessment”, which assess the threats consequences and their severity level.

In particular, SGAM extensions and interfaces have been developed to allow relevant information to be exported from the SGAM framework to other tools used in the risk management process, including the SPARKS cybersecurity simulation environment reported in [Pro16]. Additionally, an ontology-based tool has been implemented to automatically generate attack graphs, using information extracted from the SGAM framework. Finally, results of theoretic analysis of simplified use cases have been implemented in a numerical computation tool for obtaining preliminary impact assessment results, which may be used to narrow down the relevant attack patterns for detailed analysis through simulation environments such as the one developed in [Pro16].

Although the developed tools help support the implementation of the risk management process for smart grids, there are still additional features that are desirable in such a tool-chain. Firstly, the use case description currently supported by SGAM does not include a detailed characterization of the physical grid topology and the monitoring and control algorithms implemented in DER devices and at the control center. Thus, the integration of the SGAM framework and the system analysis tools and simulation environment requires a complex extension of the SGAM modelling capabilities or, alternatively, the integration with other modelling tools. Secondly, the tool-chain consists of multiple domain-specific tools, which may require substantially different expertise to be used in one same risk assessment exercise. Although a single overarching tool suite could be desirable, it would require substantial time and effort to develop such a tool suite, particularly given the objective of using open-source software.

In the future, we plan to apply the risk management process and the developed tool-chain to perform a detailed risk assessment for one of the SPARKS testbeds.
References


A System Ontology

This section gives the SPARKS ontology definition for SGAM. It serves as specification of the interface between the SGAM modeling tool and the machine-based threat and vulnerability analysis. Note that all statements that define the ontology need not be generated by the tool but are supposed to be already part of the knowledge base. Typical output of the modeling tool is given as examples, marked by using the `ex:` namespace. Examples are given in Section 3.1.

A.1 Description language and conventions

In an ontology, every entity is denoted as a resource. In this ontology, the following languages are used:

- the Resource Description Framework (RDF) to make statements about resources,
- the Resource Description Framework Schema (RDFS) as a (hierarchical) structuring schema, and
- the Web Ontology Language (OWL) as a knowledge representation language supporting model theoretic formal semantics.

In this ontology the Turtle RDF Triple notation is used as description language. A Turtle triple statement is a sequence of subject, predicate, and object, separated by whitespaces and terminated by a dot. The terminating dot at the end of the triple signalizes that the statement is complete. A typical example statement is the following triple, where Document is the subject, has the predicate, and Author the object.

```
Document has Author .
```

Triples can be written in a more compact form by the use of comma (the next statement uses the same subject and predicate as this statement) and semicolon (the next statement uses the same subject as this statement), e.g.,

```
Sparks hasMember AISEC .
Sparks hasMember AIT .
```

could also be written as

```
Sparks hasMember AIT ;
 hasMember AISEC .
```

or

```
Sparks hasMember AISEC, AIT .
```

The combined use of comma and semicolon is valid too:

```
Sparks hasMember AISEC ,
 AIT ;
 hasGrantNumber 608224 .
```
In these examples, Sparks is the subject, hasMember and hasGrantNumber are predicates (properties), while AIT, AISEC, and 608224 are objects.

Another form of abbreviation is the use of namespace prefixes in order to shorten the resource names. Resources are given as URI, without namespaces, the document would be more difficult to read for humans. The following prefixes are used at this document:

@prefix : <http://www.semanticweb.org/owl/owlapi/turtle#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix sgam: <http://www.project-sparks.eu/sgam#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix ex: <http://example.org/> .

The complete Turtle RDF triple language description can be found at http://www.w3.org/TR/turtle/.

RDFS and OWL class relations: The following predefined [RDFS] and [OWL] classes are used in combination with the listed properties to define a class hierarchy and to instantiate individuals:

- Classes are instances of owl:Class.
- Properties are instances of owl:ObjectProperty.
- rdf:type or simply a indicates that the subject is an instance of the object class.
- rdfs:subClassOf is used to indicate that the subject is a subclass of the object, i.e., the superclass inherits all a (and rdf:type) relations from the subclass.
- Matches between different ontologies are expressed by owl:sameClassAs and owl:samePropertyAs. These properties indicate, that the two classes respectively properties are synonyms of each other.
- Besides their assigned type, Instances are also instances of owl:NamedIndividual.

Naming convention: In this document, the following naming convention for classes, relations, and instances will be used:

- Classes are written in CamelCase (e.g. MyClass).
- Relations are written in mixedCase (e.g. isRelatedTo).
- Instances contain an underscore and optionally a sequential-number (e.g. Instance_1, or Instance_One)

A.2 System Ontology Description

A.2.1 Component Layer

All components within the smart grid context are located at the [SGAM] component layer.

sgam:Component  rdf:type  owl:Class ;
                 rdfs:subClassOf  owl:Thing .
**Placements:** In [SGAM](http://example.com) layers, domains and zones are used as dimensions for content specification. As the layer of a resource is implicitly given by the class type of the resource, only explicit classes for zones and domains are given:

```
sgam:Zone rdf:type owl:Class .
sgam:Domain rdf:type owl:Class .
```

The following zones are defined by SGAM:

```
sgam:MarketZone rdf:type sgam:Zone, owl:NamedIndividual .
sgam:EnterpriseZone rdf:type sgam:Zone, owl:NamedIndividual .
sgam:OperationZone rdf:type sgam:Zone, owl:NamedIndividual .
sgam:StationZone rdf:type sgam:Zone, owl:NamedIndividual .
sgam:FieldZone rdf:type sgam:Zone, owl:NamedIndividual .
```

The following domains are defined by SGAM:

```
```

The operation and placement of each object are described by its zone and domain. With the following properties, components can be assigned to zones and domains:

```
sgam:inZone rdf:type owl:ObjectProperty ;
      rdfs:domain sgam:Component ;
      rdfs:range sgam:Zone .
sgam:inDomain rdf:type owl:ObjectProperty ;
      rdfs:domain sgam:Component ;
      rdfs:range sgam:Domain .
```

**Type of component:** By deriving subclasses of sgam:Component, the type of equipment can be specified. As components could be assigned to either IT equipment or power equipment, these are defined as two basic subclasses of components.

```
sgam:ITEquipment rdfs:subClassOf sgam:Component .
sgam:PowerEquipment rdfs:subClassOf sgam:Component .
```

Subclasses of sgam:ITEquipment contain control devices like computers, or communication network equipment like networks, switches, and routers.

**Computers:** Computers are components which are further classified into servers, workstations and embedded systems. The subclass relation is used to refine the type of hardware used. The operating system and other software are described in a different manner, as shown later.
sgam:Computer rdfs:subClassOf sgam:ITEquipment.
sgam:Server rdfs:subClassOf sgam:Computer.
sgam:Workstation rdfs:subClassOf sgam:Computer.
sgam:EmbeddedSystem rdfs:subClassOf sgam:Computer.

Further subclasses can (and should) be specified here as needed. For example a PLC class could be defined as:

sgam:ProgrammableLogicDevice rdfs:subClassOf sgam:EmbeddedSystem.

Note further, that a class might be subclass also of several classes, i.e., an intelligent sensor could be both, IT equipment and power equipment.

**CommunicationEquipment:** The CommunicationEquipment subclass comprises all kind of network devices like routers, switches, etc.

sgam:CommunicationEquipment rdfs:subClassOf sgam:ITEquipment.
sgam:CommunicationSwitch rdfs:subClassOf sgam:CommunicationEquipment.
sgam:SignalTransmitter rdfs:subClassOf sgam:CommunicationEquipment.

**Software:** Software, which is running on a component, is specified using an own class hierarchy that inherits from sgam:Software. The property sgam:runsOn links the software to the component it is running on. Each software instance runs on a component, i.e., it needs exactly one runsOn relation.

sgam:Software rdf:type owl:Class.
sgam:OperatingSystem rdfs:subClassOf sgam:Software.
sgam:ApplicationSW rdfs:subclassOf sgam:Software.

sgam:runsOn rdf:type owl:ObjectProperty;
  rdfs:domain sgam:Software;
  rdfs:range sgam:Component.

A valuable information about software is the version number. As there is no uniform version information syntax, it is possible to provide a string to describe the version.

sgam:isVersion rdf:type owl:DataProperty;
  rdfs:domain sgam:Component, sgam:Software;
  rdfs:range xsd:string.

Patches are another subclass of software, which can be applied on other software.
CommunicationNetwork: CommunicationNetwork is the subclass of Component that comprises the networks in the component layer (i.e. the physical infrastructure). It can be refined into several subclasses.

A SwitchedCommunicationNetwork is a special subclass, that considers networks that are managed by a switch or another managing device (i.e. bus master). Using this class together with the switchedBy property avoids that in a switched network, every cable has to be modelled individually.

An example for multiple inheritance is a network with the properties of a wired network including a switch, making it a switched network. This leads to a subclass that inheritances from WiredCommunicationNetwork as well as SwitchedCommunicationNetwork.

Example 10. A PC that is connected via a switched Ethernet network to a PLC can be modeled as:
Power Equipment: Power equipment can be modelled in a similar fashion as computers and network equipment. As this kind of hardware is not required for the threat analysis and the SGAM does not provide a detailed hierarchy either, only basic examples are provided to showcase how such components could be covered in this ontology.

Mandatory Information: Although, each information improves the quality of possible statements, it must be accepted that the available amount of information about systems is typically very limited. While, for example, the placement information \texttt{inDomain} and \texttt{inZone} are not required for functional analysis, existing network connections are.

In summary, a minimum amount of knowledge, including the following things and properties is required:

- Components: All present physical components of a system must be known (at least at a box level like a PC, a router, etc.).
  - label: a label to refer the component.
  - connectedTo: if the component has a communication connection to a network

- Computer: For computers, it is necessary for our analysis, that at least their type is provided.
  - Class: Server, Workstation or EmbeddedSystem

- Software: For software, at least the type and function (like "setpoint calculation software") are necessary.
  - Class: OperatingSystem, ApplicationSW, etc.
  - runsOn: the component where the software instance is running

- CommunicationNetwork: To see the components interconnected by the network, all networks must be provided. This also serves as an aggregator for the connectedTo property of a component. While the component information could be extracted from the SGAM, the subClass assignment is also required.

- Further, it is necessary to know the subClass of the existing PowerEquipment and CommunicationEquipment.
A.2.2 Communication Layer

Protocols and mechanisms for the exchange of information between components are described by the communication layer.

sgam:Protocol rdf:type owl:Class .

Protocols are used to connect components via a communication network and could be stacked (e.g. TCP/IP requires IP) by the usesProtocol relation.

sgam:usesProtocol rdf:type owl:ObjectProperty ;
    rdfs:domain sgam:Protocol ;
    rdfs:range sgam:Protocol .

A protocol requires a network to operate on. As a convention, it is sufficient to link only the lowest protocol of a stack to the network.

sgam:usesCommunicationNetwork rdf:type owl:ObjectProperty ;
    rdfs:domain sgam:Protocol ;
    rdfs:range sgam:CommunicationNetwork .

Protocols provide connections between software. This could be the case for different software within one computer as well as connections between software instances at computers that are multiple network hops away. By providing a connects relation between two distant software applications, logical connections become obvious and potential routes get highlighted. The software is also an endpoint for each connection.

sgam:connects rdf:type owl:ObjectProperty ;
    rdfs:domain sgam:Protocol ;
    rdfs:range sgam:Software .

Note that a protocol connects to sgam:Software and not sgam:Component. Therefore, for every protocol endpoint, at least one software (e.g. the OS) needs to be specified.

Mandatory Information: For the communication layer, all communication networks and protocol connections (between applications) must be provided. This could be extracted directly from the SGAM model.

Example 11. Two applications of different components are corresponding via GOOSE across two IP routed networks.

ex:PC_1 rdf:type sgam:Workstation ;
    sgam:inDomain sgam:DerDomain ;
    sgam:inZone sgam:OperationZone ;
    sgam:runsSW: ex:CNC_app_1 .

ex:PLC_1 rdf:type sgam:PLC ;
    sgam:inDomain sgam:DerDomain ;
    sgam:inZone sgam:ProcessZone ;
    sgam:runsSW: ex:PLC_sw_1 .
ex:Router_1 rdf:type sgam:CommunicationEquipment ;
  rdfs:comment "Router that connects the two networks." ;
  sgam:inDomain sgam:DerDomain ;
  sgam:inZone sgam:ProcessZone .

ex:Eth_1 rdf:type sgam:CommunicationNetwork ;
  rdfs:comment "Network at the control centre" ;
  sgam:inDomain sgam:DerDomain ;
  sgam:inZone sgam:OperationZone .

ex:Eth_2 rdf:type sgam:CommunicationNetwork ;
  rdfs:comment "Network for the DER equipment at the process zone" ;
  sgam:inDomain sgam:der_domain ;
  sgam:inZone sgam:process_zone .

ex:IP_connection_1 rdf:type sgam:Protocol ;
  sgam:usesCommunicationNetwork ex:Eth_1 ;
  sgam:connects ex:OS_1, ex:Router_firmware_1 ;
  rdfs:label "IP_PC1_Router" ;
  rdfs:comment "IP connection between PC_1 and Router" .

ex:IP_connection_2 rdf:type sgam:Protocol ;
  sgam:usesCommunicationNetwork ex:Eth_2 ;
  sgam:connects ex:OS_1, ex:Router_firmware_1 ;
  rdfs:label "IP_PLC1_Router" ;
  rdfs:comment "IP connection between PLC and Router" .

ex:TCP_connection_1 rdf:type sgam:Protocol ;
  rdfs:label "The TCP connection between the PC and PLC networks" ;
  sgam:usesProtocol ex:IP_connection_1, ex:IP_connection_2 ;
  sgam:connects ex:CC_app_1, ex:PLC_sw_1 .

ex:Goose_connection_1 rdf:type sgam:Protocol ;
  sgam:usesProtocol ex:TCP_connection_1 ;
  sgam:connects ex:CC_app_1, ex:PLC_sw_1 .

ex:OS_1 rdf:type sgam:OperatingSystem ;
  rdfs:comment "WindowsXP running at PC_1" ;
  sgam:runsOn ex:PC_1 .

ex:Router_firmware_1 rdf:type sgam:OperatingSystem ;
  rdfs:comment "Firmware of the router" ;
  sgam:runsOn ex:Router_1 .

ex:CC_app_1 rdf:type sgam:ApplicationSW ;
  rdfs:comment "Command and Control Software running at PC_1" ;
  sgam:runsOn ex:PC_1 .
A.2.3 Information Layer

The information layer contains the description of all (shared) information objects, their local representation, and their interdependencies.

A SharedInformationObject is a data object (data model) which is shared between different components.

A LocalInformationObject is the local representation of a SharedInformationObject, which is stored at a specific component and could be an output of the modelled system. When using the isOutput relation, it is important to pay attention to when the model is extended and the former output target is taken into the model.

A LocalInformationObject can hereby act as a reader or writer (or both) of the SharedInformationObject. Protocols are used to implement the data exchange inside the shared object:

Example 12. Exchange of a setpoint data object between two components. The connection of the protocol to the components’ software is omitted here.

ex:Sender_component_1 rdf:type sgam:Component ;
   rdfs:comment "Component sending the setpoint" ;
   sgam:Component

ex:Receiver_component_1 rdf:type sgam:Component ;
   rdfs:comment "Component sending the setpoint" ;
   sgam:Component
LocalFunctions represent inter-dependencies between information objects. They are implemented at a component and operate on LocalInformationObjects. With SharedInformationObjects, LocalInformationObjects, and LocalFunctions, it is possible to model the dependency of information objects.

Mandatory Information: Each asset relevant information as well as the affecting LocalFunctions must be identified and assigned to a component and network. This could be extracted directly from the SGAM model.

Example 13. LocalFunction example
A.2.4 Function layer

The function layer describes functions and services including their relationships from an architectural viewpoint. Functions depend on the LocalInformationObjects at specific component.

\[
\text{sgam:Function } \text{rdf:type } \text{owl:Class}.
\]

\[
\text{sgam:dependsOnInformation } \text{rdf:type } \text{owl:ObjectProperty} ; \\
\text{rdfs:domain } \text{sgam:Function} ; \\
\text{rdfs:range } \text{sgam:LocalInformationObject}.
\]

**Mandatory Information:** For the function layer, each critical function and the information it depends on must be referenced.

B Preliminary Concepts for System Analysis

In this section, we recall important properties of certain classes of linear time-invariant (LTI) systems that will be useful in the subsequent sections. Let us consider a general LTI system of the form:

\[
\begin{aligned}
\dot{x}(t) &= Ax(t) + Fu(t) \\
y(t) &= Cx(t) + Du(t),
\end{aligned}
\]  

(19)

where \(x(t) \in \mathbb{R}^n\), \(u(t) \in \mathbb{R}^m\) and \(y(t) \in \mathbb{R}^p\) are the system state, the control input, and the controlled output at time \(t\), respectively. Denoting \(a_{ij} = [A]_{i,j}\) as the entry of \(A\) in the \(i\)-th row and \(j\)-th column, the class of diagonally dominant matrices is defined as follows.

**Definition 3** (Diagonally dominant matrices). The matrix \(A\) is said to be row-diagonally dominant if its entries satisfy the conditions

\[
|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|, \quad \forall i \in \{1, \ldots, n\}.
\]  

(20)

Given the above definition, the system (19) is said to be row-diagonally dominant if the state matrix \(A\) is row-diagonally dominant.

Another relevant class of systems is that of positive systems (see [Ran15], for instance), which play an important role throughout this paper.

**Definition 4** (Positive systems). The LTI system (19) is said to be positive if the following conditions hold:

1. The matrix \(A\) is Metzler, i.e., it has non-negative off-diagonal entries;
2. The matrices \(F, C\) and \(D\) are non-negative, i.e., they only have non-negative entries.

Positive systems have several interesting properties, e.g., \(x(0) \geq 0\) and \(u(t) \geq 0\) result in trajectories satisfying \(x(t) \geq 0\) for all \(t\), where \(x \geq 0\) denotes element-wise inequalities. In particular, the following properties of positive systems are instrumental in our analysis.

**Lemma 3** ([Ran15]). If the system (19) is positive, the following statements hold:

1. The matrix \(A\) is Hurwitz (every eigenvalue of \(A\) has strictly negative real part) if, and only if, there exists a \(\xi \in \mathbb{R}^n\) such that \(\xi > 0\) and \(A\xi < 0\).
2. Let \( m = p = 1 \), define \( H(s) = C(sI - A)^{-1}F + D \) as the transfer function of the system (19), and suppose \( A \) is Hurwitz. The \( L_{\infty} \)-induced norm of (19) is given by

\[
\| H \|_{L_{\infty}\text{-ind}} = \sup_{\| u \|_{L_\infty} < \infty} \frac{\| y \|_{L_\infty}}{\| u \|_{L_{\\infty}}} = H(0). \tag{21}
\]

The first property relates the stability of positive systems to a set of inequality constraints, which is helpful to derive stability conditions. Regarding the input-output behavior of the system, the second property characterizes the maximum amplitude of the output signal \( y(t) \) that can be achieved by an input \( u(t) \) with bounded amplitude

\[
\| u \|_{L_{\infty}} := \sup_{t \geq 0} |u(t)| < \infty.
\]

In particular, constraining the input’s amplitude to be at most 1, i.e., \( \| u \|_{L_{\infty}} \leq 1 \), the corresponding output satisfies the tight inequality \( \sup_{t \geq 0} |y(t)| \leq H \| u \|_{L_{\infty}} \), which holds with equality for a constant input \( u(t) = 1 \).

Moreover, the second property leads to other relevant features of positive systems in terms of input-output behavior. On one hand, for input signals \( u(t) \) corresponding to reference signals, it establishes the absence of output overshoot for step changes in the reference. On the other hand, considering \( u(t) \) to be a possible disturbance or anomaly, the \( L_{\infty} \)-induced norm quantifies the worst-case impact that the anomaly can have on the output signal, in terms of signal amplitudes. In this work, both of these interpretations are further discussed and illustrated in the context of power systems.