SecuriDN: a Customizable GUI Generating Cybersecurity Models for DER Control Architectures

Davide Cerotti[1,*], Daniele Codetta-Raiteri[1], Giovanna Dondossola[3], Lavinia Egidi[1], Giuliana Franceschinis[1], Luigi Portinale[1], Davide Savarro[2] and Roberta Terruggia[3]

[1] *Computer Science Institute, DiSIT, Università del Piemonte Orientale (UPO), 15121 Alessandria, Italy*

[2] *Computer Science Department, Università di Torino, 10149 Torino, Italy*

[3] *Transmission and Distribution Technologies Department, Ricerca sul Sistema Energetico (RSE), 20134 Milano, Italy*

Abstract

SecuriDN is a tool for the representation of the assets composing the IT and the OT subsystems of DER (Distributed Energy Resources) control networks and the possible cyberattacks that can threaten them. In this paper the main goals of such tool and its features are described using a simple example.

Keywords

Early evidence-based cyberattack detection, cyber threats, power systems, Distributed Energy Resources, Bayesian Networks, time-driven attack analysis, multiformalism models

1. Introduction

Cybersecurity of critical systems is receiving a great attention in the endeavour to guarantee national safety. Cyberattacks to such systems can have global and disastrous consequences. We focus in particular on the *electro-energetic sector*, which in recent decades is following a trend of increasing complexity, transitioning from a unidirectional "producer to consumer" model to a grid in which each actor can be both producer and consumer. The coordination and optimisation of the generated/consumed fluxes requires a continuous exchange of information and therefore a high level of connectivity of all involved devices. This in turn provides an environment rich of opportunities for adversarial activity, since complexity tends to bring along vulnerability. Several European and national legislative acts address the data exchanges of energy infrastructures and their cybersecurity. The European Union Regulation 2017/1485 SOGL System Operation Guideline aims to provide a set of guidelines including operation security for transmission grid, harmonised rules for transmission and distribution system operators and Significant Grid User (SGU) for interconnection operations. The EU cybersecurity Directive

ITASEC 2024: The Italian Conference on CyberSecurity, April 08–12, 2024, Salerno, ITALY

*Corresponding author.

davide.cerotti@uniupo.it (D. Cerotti); daniele.codetta@uniupo.it (D. Codetta-Raiteri); giovanna.dondossola@rse-web.it (G. Dondossola); lavinia.egidi@uniupo.it (L. Egidi); giuliana.franceschinis@uniupo.it (G. Franceschinis); luigi.portinale@uniupo.it (L. Portinale); davide.savarro@unito.it (D. Savarro); roberta.terruggia@rse-web.it (R. Terruggia)

0000-0001-5192-9387 (D. Cerotti); 0000-0001-8881-2537 (D. Codetta-Raiteri); 0009-0001-0387-4374 (G. Dondossola); 0000-0002-9745-0942 (L. Egidi); 0000-0001-6571-9217 (G. Franceschinis); 0000-0002-6053-4667 (L. Portinale); 0009-0002-7247-0346 (D. Savarro); 0000-0003-0345-7332 (R. Terruggia)

NIS, introduced in 2016, has been updated with NIS2 Directive which came into force in 2023 and introduces new cybersecurity obligations for essential and important critical infrastructure operators. Each member state including Italy implements the NIS2 Directive through a series of national legislative acts. Moreover specific national norms of the energy sector include cybersecurity obligations based on international standards, as for example CEI 0-16 (high and medium voltage SGU connection rules) and CEI 0-21 (low voltage SGU connection rules).

In this setting, we are working on the development of a set of tools to enable the cybersecurity analyst to face the threats in a more informed way, requiring fewer competences on the side of the specialised personnel. It is well known that cyberattacks progress through various steps of a cyber kill-chain, and our objective is to enable early detection of threats, in order to be able to discover adversarial activity in its first phases of the process to prevent any damage to the assets and any degradation of functionality.

In this paper we present a graphical tool, called SecuriDN[1], that enables the security analyst to define AI based detection models by just defining, through a graphical interface, the network architecture of interest. The user combines predefined assets and defines their relationships according to the existing architecture. Each resource that intervenes in this construction contains information on the attacks that can be made to the resource itself, and their relations indicate how such attacks can propagate to other assets. When the architecture is complete, SecuriDN combines the information present on the various nodes into an attack graph that represents the possible attack processes that the adversary can choose. Since in realistic complex systems manual generation of the attack graph can be time-consuming and error-prone, its automatic production is a first valuable support to the security analyst's work.

Moreover, the tool is capable of deriving from the attack graph a Dynamic Bayesian Network (DBN) used to compute the probability that a specific target is or will be compromised. In the field of artificial intelligence, Bayesian Networks are graphic-probabilistic models that are widely used for the representation of uncertain knowledge and in particular DBNs are able to model the temporal aspects, useful for capturing the dynamics of an attack process. Furthermore, the formalism allows exploiting the dependencies between the modeled entities to significantly reduce the number of parameters necessary to specify the stochastic behavior of the process. The DBN's parameterization, out of the scope of this paper, can be achieved in two possible ways in our system: using measurements observed in a real system or experimental testbed; or learning from data extracted from a simulation model.

The SecuriDN tool is a component of a larger platform that provides a complete architecture for a flexible detection system of adversarial activity. SecuriDN runs on the cyber security analyst's workstation and enables the design of new models for analysis of the evidences. Such models will be deployed as detection modules in the framework. Evidences collected from the monitored network are filtered in real time by an Opensearch database pipeline and provided to the detection modules through a communication channel implemented according to a producer-consumer scheme to enable coexistence and cooperation of different modules at the same time. The results of the analyses performed by the modules is then presented to the analyst via the Opensearch Dashboard, enabling detection of adversarial activity while it is taking place. The platform is currently part of a testbed for detection models [1]. It collects information from an

[1]The name SecuriDN derives from the use of *Draw-Net* as graphical user interface (Sec. 3.2).

emulated network on which a synthetic adversary executes attack processes. In this setting, since we control also the adversarial activity, we can test the performance of the detection models.

Since SecuriDN is capable of working with multiple formalisms, it enables to collect in a single tool the experiences of heterogeneous experts that can then be leveraged using a simple graphic interface. The diversification possible thanks to this multiformalism approach enables the selection of the most appropriate formalism, possibly customized to tackle more precisely the specific detection problem.

The paper is organized as follows. In Sec. 2 we discuss related work. Sec. 3 provides necessary preliminary notions. SecuriDN is presented in Sec. 4. Finally we conclude the paper in Sec. 5.

2. Related work

There has been a significant amount of work on developing quantitative evaluation tools for computer security. In particular, the methodology of *Attack Trees* (AT) has become popular and has been applied in several contexts, such as SCADA systems [2, 3, 4]. In particular, in an AT, attacks against a system can be represented in a tree-like structure, with the goal as the top node and different ways of achieving that goal as multi-level hierarchical structures based on logical (i.e. Boolean) AND/OR operators (gates). Leaves on this hierarchy represent *basic attacks*; these are specific operations an attacker can put in place, in order to pursue his ultimate goal. Inner nodes are *non-basic attacks* (i.e. consequences of attacks) and are defined in terms of Boolean combination of events.

Since the basic AT does not include any defense mechanism, extensions have been proposed to incorporate defense mechanisms (or countermeasures) [5] and other features [6]. In these cases, they are also known as *Defence Trees* (DT) [7]. In [8], the point of view of the attacker as well as the point of view of the defender can be analyzed. Besides ATs, other modelling formalisms have been applied to security. *Petri Nets* (PN) are exploited for penetration testing in [9], and for quantification of risk of attack to SCADA systems in [10].

In general, AT models are easy to build and very readable, but they lack modelling power because they can only represent the features that Boolean gates can express. PN based models can model more sophisticated events, but they are harder to build, less readable and less intuitive to interpret. A trade-off is the generation of PN models from AT [11, 12]. In this way, the attack can be easily represented with a familiar model like ATs, and the corresponding PN can be automatically generated, and possibly edited to include further aspects that ATs cannot capture. Other modelling formalisms are *Privilege Graphs* [13] and *Attack Graphs* (AG) [14] where the model of the attack must not follow a tree structure as in ATs.

An example of a security assessment tool that combines some of the previously mentioned formalisms is ADVISE (*ADversary VIew Security Evaluation*) [15, 16], where an enriched AG (containing time, cost, probability and outcomes of the attack steps) is coupled with a specific adversary profile (including attacker's skills, goals and preferences) to generate a State Look-Ahead Tree (SLAT) that represents a subset of all the possible attack scenarios reachable by the attacker. A further extension of this framework called the ADVISE Meta Model Formalism [17] is used to ease the creation of ADVISE models starting from a higher level specification of the

system resources and adversary profiles. In comparison to ADVISE, the current implementation of SecuriDN supports a limited level of characterization of adversarial behavior by modelling different skill levels of the attackers with different mean times to completion of attack steps. ADVISE doesn't model evidence, and therefore it is not suited for real-time detection. In contrast, SecuriDN supports analytic-based predictive models that can be used for online detection activities. Moreover, it has been specifically designed to be integrated in a modular detection platform, into which the generated custom detection models can be automatically deployed.

Bayesian Networks (BN) are a widely used formalism for representing uncertain knowledge in probabilistic systems, applied to a variety of real-world and complex problems. The adoption of BNs, and their temporal aware versions, *Dynamic Bayesian Networks* (DBN), for security modeling has been advocated by several researchers [18, 19, 20, 21]. Such approaches start from AG models to show how BNs can be derived, stressing the evidence-based analysis allowed by such a formalism.

SecuriCAD [22] is a security assessment tool that allows to define the architecture of ICT infrastructures and automatically generates the AG according to a predefined library of attack steps. Models can be defined using a Domain Specific Language (DSL) expressed in the Meta Attack Language (MAL) [23]. The resulting AG is evaluated by Monte Carlo simulation to compute the distribution of the mean completion time of the identified critical paths. In comparison, SecuriDN allows to define the architecture and customize the attack process of each asset in a graphic version of a MAL-based DSL. It also generates an AG, which is translated in a DBN capable to capture the temporal evolution of the attack. Its evaluation provides results conditioned on a stream of observations. These features make the model suitable to be used as an online detection module able to adjust its results learning from the collected evidences.

3. Preliminary notions

3.1. Energy systems' cybersecurity

The cybersecurity of energy infrastructures has increasingly become a relevant field of study that has attracted attention from both industry and academic world. This interest has grown along the years with cyber incidents that have occurred in the recent past, such as Stuxnet [24] and CrashOverride/Industroyer [25]. The first one targeted Iranian nuclear power plants and, by exploiting communication through the programmable controller, it successfully reconfigured some operational parameters of the centrifuge units. The second one instead targeted some Ukrainian distribution grids and, among its other capabilities, it could send malicious control commands directly to Remote Terminal Units (RTU) to toggle circuit breakers in a rapid open-close-open-close pattern and caused a widespread blackout. This kind of cyberattacks were made possible by the capability to compromise IT/OT networks infrastructures based on TCP/IP technologies. The energy transition that will characterize the next future requires a pervasive digitalisation of the infrastructure. This increases the cyber risk in terms of extension of attack surface. This new landscape requires new functionalities and systems for the operation of power infrastructures, including control centers, substations, generation plants and loads. In particular distributed energy resources connected in medium and low voltage, especially

generation resources from renewable sources and electric vehicle charging infrastructure are characterized by an unpredictable power profile. The operation of the infrastructure including these components, the management of the flexibility of the power demand and the need to provide ancillary services to grid operators requires secure ICT infrastructures.

An example of this kind of trend can be observed in the infrastructure of modern substations [26]. Power transformation substations are in fact responsible for managing the voltage/current transition from transmission grids (e.g. $66kV$) to distribution grids (e.g. $11kV$). Human Machine Interface (HMI) and a Supervisory Control & Data Acquisition (SCADA) system are typical nodes of the control architecture. These allow to monitor the status of the power grid and initiate control operations. Within this complex framework it is essential to adopt a communication standard to ensure interoperability between various devices and vendors. For this purpose, the IEC 61850 [27] standard has been defined, in fact it specifies both the data model underlying the substation and its mappings onto various communication protocols. Some of them are specific to perform station-bus communication (mainly SCADA/HMI querying IEDs and PLCs) such as the Manufacturing Message Specification (MMS) that works over TCP/IP and for this reason it is vulnerable to cyberattacks such as Man In The Middle (MITM) or Distributed Denial of Service (DDoS). Others are responsible for process-bus communication such as the Sampled Values (SV) and the Generic Object Oriented Substation Event (GOOSE) protocols. The SV protocol is used to carry digitalized measurements taken from physical devices to the remote IED, while the GOOSE protocol has been introduced for announcing status updates across various IEDs (e.g. open or closed state of a circuit breaker controlled by a certain IED). These last two protocols are more time critical so they function directly upon link-layer communication, but they can still be target of False Data Injection attacks (FDI) [28].

In order to defend against this type of cyber assaults, the IEC 62351 series of standards [29] was developed, but the security measures it introduces (e.g. digital signatures and authentication schemes) are not often implemented in brownfields due to legacy issues related to existing equipment and applications [30]. As specified by the Norm CEI 0-16 for the new applications performing DER control, in the focus of this paper, the implementation of cybersecurity profiles compliant with IEC 62351 is mandatory.

3.2. Draw-Net Modeling System

The design of complex systems can be fruitfully supported by modeling: both qualitative and quantitative measures can be evaluated on the models, and the results can be used to guide the design. Models are the basis of Model Driven Engineering (MDE) techniques [31], and it is very important to pursue the goal of embedding in a single flexible framework the possibility of choosing among multiple modeling formalisms and solution methods, in order to represent and evaluate the system by means of the most suitable model and solver. Software tools for performance and dependability analysis have been developed with this goal in mind, such as *Möbius* [32] and *SHARPE* [33], but the set of supported formalisms is usually predefined and closed.

The Draw-Net Modeling System (DMS) [34, 35, 36] is a customizable framework supporting the design and the solution of models expressed in any graph-based formalism. The system is characterized by an open architecture and includes an XML based language family that can

be used to define existing as well as new formalisms and the models expressed through such formalisms. The original idea behind DMS, that differentiates it from the other approaches, is the possibility of easily adding new formalisms without recompiling the DMS source code and the fact that it favours the reuse and integration, with a small programming effort, of existing tools for solving models.

During the years, many formalisms (Petri Nets, Bayesian Networks, Fault Trees, etc.) and the corresponding solvers have been included in DMS [37, 38, 39, 40, 41, 42].

3.2.1. DMS general architecture

DMS is a Java-based framework exploiting the *DNlib* library [34, 35]. The general architecture of DMS is composed by the following main levels (Fig. 1).

The formalism level defines all the primitives that can be used to design a model. A formalism is defined as the tuple $F = \{E, P, C, S, H, T_P\}$ where E is the set of *Elements*; P is the set of *Properties*; C is the set of *Constraints*; S is the *structure function* associating each element to its properties; H is the *inheritance function* setting that one or more elements inherit the properties of a specific (abstract) element; T_P is the *property typing function* setting the type of each property.

Elements correspond to the possible nodes and arcs in the model. *Properties* are the attributes associated with an element. Moreover, an element has graphical properties (shape, size, color, etc.). Properties are typed: they can only contain values of a specific type (integer, float, string, Boolean, etc.). Finally, *Constraints* are logical propositions that describe required consistency relations among elements and properties of a model.

XML files contain the elements, their properties (including the graphical ones), and the solver(s) associated with the formalism.

The model level describes a system using the primitives defined in the formalism to specify a model which is defined by the tuple $M = \{F, I, m_0, T, V, L\}$ where F is the formalism; I is the set of element instances (every $i \in I$ represents an instance of an element of F); $m_0 \in I$ is the *main* element; T is the *element typing function* associating $i \in I$ with the formalism element to which i corresponds (the element must not be abstract); V is the *assignment function* which specifies the property values ($V(i, p)$ is the value of property p of instance $i \in I$).

The user exploits *Draw-Net* to select a formalism among the available ones, load its definition from the XML files, and design models conforming that formalism (Fig. 1).

The solver level concerns the conversion, the analysis, the simulation, or any other elaboration of the model. Still by means of *Draw-Net* the user can set the results to compute, save the model into one XML file, and execute the solver on the model. The results produced by the solver can be shown by *Draw-Net* at the end of the model solution (Fig. 1).

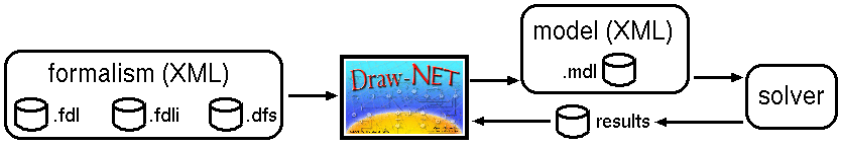


Figure 1: DMS general architecture

3.3. Dynamic Bayesian Networks

Bayesian Networks (BN) [43, 44] are the most adopted formalism for uncertain reasoning. A BN is a directed acyclic graph whose nodes correspond to discrete random variables that have a conditional dependence on the parent nodes, with probabilities defined via Conditional Probability Tables (CPT). For nodes with no parents unconditional probabilities are defined. Dynamic Bayesian Networks (DBN) extend BNs by providing an explicit discrete temporal dimension [45].

A DBN can in general represent semi-Markovian stochastic processes of order $k-1$, providing the modeling for k time slices. When the Markovian assumption holds ($k = 2$), only 2 time slices are considered in order to model the system temporal evolution: the slice at time t depends only on the previous slice at $t - \Delta t$, and is conditionally independent of the past ones. In such a case we have a two time slice temporal Bayesian Network (a 2-TBN, see Appendix A).

In our setting the nodes of a DBN represent either attack steps or evidence collected from the monitored network by the analytics. The DBN enables inference of the security posture of the network through the observed evidence. Several kinds of inferences can be carried out using a DBN, supporting the analyst's decision process in various ways:

- Monitoring of the security posture: it is possible to compute in real time the probability that an attack attempt is taking place, based on the observations gathered by the analytics. When this probability value exceeds a given threshold, the analyst can infer that an attack attempt is in progress. This feature can support early detection.
- Prediction of adversarial activity: the predictive inference task allows to identify the subset of techniques that the attacker will more likely exploit in the future, again based of evidence obtained from the analytics. This allows the analyst to plan ahead, setting up appropriate defensive actions.
- Diagnosis: DBNs also help understand the causes of security events. The diagnosis can be carried out in real time ("what is happening?") or it can be a deeper *post-incident* investigation. This feature supports revision of security and monitoring decisions.

Moreover, when security measures are also modelled through the DBN, the inference tasks can enable assessment of the effectiveness of such measures. The temporal dimension of DBNs allows to process streams of evidence maintaining the above support information up to date. A higher level of confidence of such data on the part of the analyst is possible because DBNs guarantee *explainability*, i.e. it is possible to understand the reasoning behind the conclusions drawn by the model. This is in contrast to the black box character of other AI models.

Different algorithms, either exact or approximate, can be exploited in order to implement inference tasks in DBN [45, 46, 47, 48].

4. The SecuriDN tool

We implemented a prototype of the tool called SecuriDN, which allows us to define the architecture of an IT/OT system, the attacks affecting the system, and the parameters that characterize the various attack steps. Given these definitions, the tool generates models that allow us to analyze the behavior of the system. The implementation of SecuriDN is based on DMS (Sec. 3.2) which provides the graphical user interface (GUI) and a library (*DNlib*) for model construction and manipulation.

SecuriDN allows us to model these aspects:

- the ICT architecture consisting of a set of assets (hosts, networks, applications, etc.), relationships between assets, the asset where the attack begins, the asset which is the goal of the attack.
- The possible attacks affecting each asset: the attacks are made up of multiple techniques combined with each other. For each asset, an Attack Graph (AG) is defined, modeling these combinations. The AG contains a node for each technique while the edges indicate how the execution of a technique can enable subsequent techniques. In addition, the AG can contain logical operators (AND, OR) and nodes to represent defenses, analytics and the asset impairment.
- The global AG obtained by combining the AGs of the individual assets present in the architecture. The union of the various AGs takes place through shared nodes. After that, the paths from the initial point of attack to the goal are maintained, and all the other paths are removed.
- The DBN derived from the global AG.

4.1. Architecture

The first step consists in defining the architecture. The *assets* currently foreseen in the formalism represent the main possible targets of an attack in a control network: specific hardware equipment, such as *IED* and physical communication *Networks*, together with software applications, like *SCADA*, *MMS server*, *HMI*. Also logical communication *Channels*, such as TCP connections and the *Dataflow* inside them are included as possible attack goals. Fig. 2 shows in background the architecture window of SecuriDN's GUI. In this window, the assets can be chosen by the user from the model panel, located on the top right side.

The resources are connected by oriented arcs whose graphic style indicates the specific relationship (In, Connect, Cross, Execute, etc.). Such relations represent potential means of attack propagation across assets. In the background of Fig. 2 we see an arc of type *In* that goes from *IED* to *DER* to indicate that the IED is connected to the DER. We also see an arc of type *Execute* that goes from *IED* to *MMSServer* to indicate that the IED is the device that executes the MMS Server performing the DER control functions and so when an adversary compromises the IED, the attack can propagate by taking control of the MMS server running on it.

The architecture contains two special nodes: *Attacker* and *Goal*. In the background of Fig. 2 the node *Attacker* is connected to the asset *MMSServer* to indicate the asset where the attack begins and the initial technique (*spoRepMes* in the example). The node *Goal* is connected to *DER*, i.e. the renewable energy source, that is the final goal of the attack.

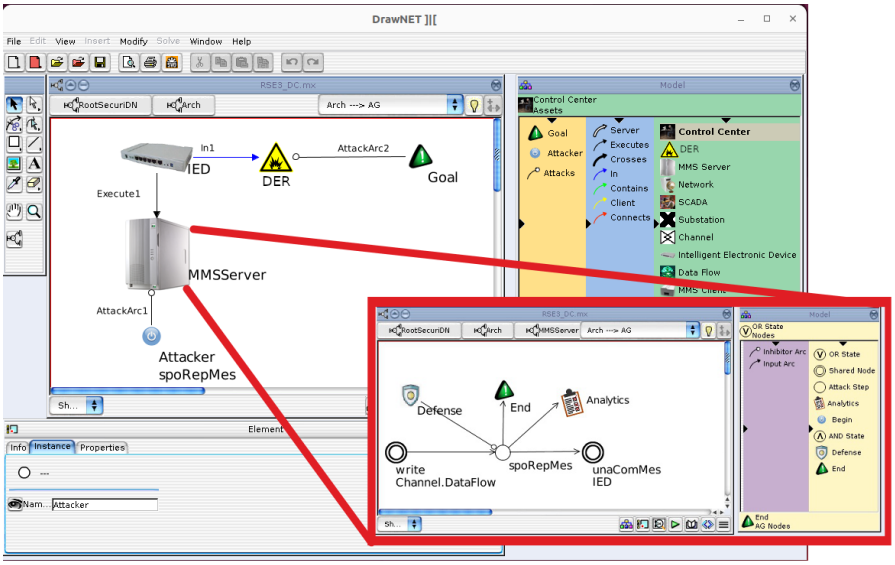


Figure 2: An example of architecture in SecuriDN with the AG of the asset *MMSServer*

In the current implementation the nodes *Attacker* and *Goal* are unique: so the attack begins from one specific asset and the final goal involves only one asset. The possibility of multiple attackers and goals may be a future development.

Notice that it is not modeled how the achievement of the final goal affects the electrical system stability, and thus the real impact of the attack. For such objective it is necessary to resort to other approaches, e.g. power flow models or power grid simulations, which in principle can be integrated with SecuriDN, but are out of the paper scope.

4.2. Local attack graphs

Each asset has a predefined local Attack Graph (lAG) as submodel (it is possible to manually add further nodes and arcs). We consider the lAG in the foreground of Fig. 2 as an example, relating to the MMS Server resource, where we see these types of node:

- an **internal technique** (simple circle) is a technique that takes place within the asset modeled by the lAG; in Fig. 2 we have *spoRepMes* (Spoof Report Message).
- an **external technique** (double circle) is a technique that takes place in another asset, but which can enable an internal technique or be enabled by an internal technique; in Fig. 2 *unaComMes* (unauthorized command message) takes place in the asset *IED*, while *Write* takes place in *Dataflow*.
- A **defense** has a graphic icon that reminds a shield,
  and is a node representing a countermeasure, such as a firewall or an antivirus, able to mitigate or even inhibit an internal technique to which it is connected.
  The mitigation degree will be reflected in the final DBN as a reduction of the technique's success probability.

In Fig. 2 we have the node *defense* modeling a generic countermeasure.

- An **analytic** is graphically represented by a sort of notepad (an analytic could be a system log); in Fig. 2 we have the node *analytics*. Security analytics describe events whose observation is significant from a security perspective, e.g. a specific item on the system log that may be a clue about the exploitation of one or more techniques.
- The **compromise** of the resource appears as a triangular signal of danger; in the example we have *End*. If in the graph relating to architecture, a certain asset is the final goal, then the compromise node in the lAG relating to the resource, corresponds to the node *Goal* in the architecture (Fig. 2) . For this reason, the node *Goal* of architecture and the compromise node of the lAG have the same graphic aspect.

In Fig. 2 we can also see the types of arc in an AG:

- An arc of **input/output** (I/O) is an oriented arc and has different roles:
  - can go from a technique (internal or external) to another technique (internal or external) to indicate that the first technique enables the second one; in Fig. 2 an arc of I/O goes from *Write* to *spoRepMes*, and another arc of I/O goes from *spoRepMes* to *unaComMes*.
  - can go from an internal technique to an analytic to indicate that the execution of the technique determines the production of system log; in Fig. 2 it is traced from *spoRepMes* to *Analytics*.
  - can go from an internal technique to a compromise node to indicate that the success of the technique determines the compromise of the asset relating to the lAG; in Fig. 2 it is traced from *spoRepMes* to *End*.
- An **inhibitor** arc goes from a defense to the technique inhibited by that defense, to specify the attack step impaired by the defense (the arc ends with a circle); in Fig. 2 it goes from *defense* to *spoRepMes*.

4.3. Automatic generation of attack models

Once the architecture is manually defined, a global attack graph and a DBN can be automatically generated with one click.

4.3.1. Global attack graph

On one hand, the generation of a global Attack Graph (gAG) is triggered. At a high level, the generation process is accomplished through the following steps:

- **Connection of lAGs:** A first, raw gAG, is built in this step: it is initially created as the union of the lAGs of all the assets in the architecture. If two assets are connected in the architecture, then the relative AGs are joined in the gAG merging shared nodes. To ensure in the gAG the uniqueness of the technique names, which may appear multiple times in different lAG, the asset name is used as a prefix of the original name.

- **Identification of attacker and goal:** According to the asset where the attack begins (node *Attacker* in the architecture) and the asset which is the goal of the attack (node *goal* in the architecture), the node relating to the initial technique and the node relating to the goal are identified in the gAG.
- **Reduction:** By visiting the gAG, the nodes and arcs belonging to the paths from the initial node to the target node are marked. All the nodes and arcs that are not marked are eliminated from the gAG, thus obtaining the final, simplified gAG (Fig. 3a).

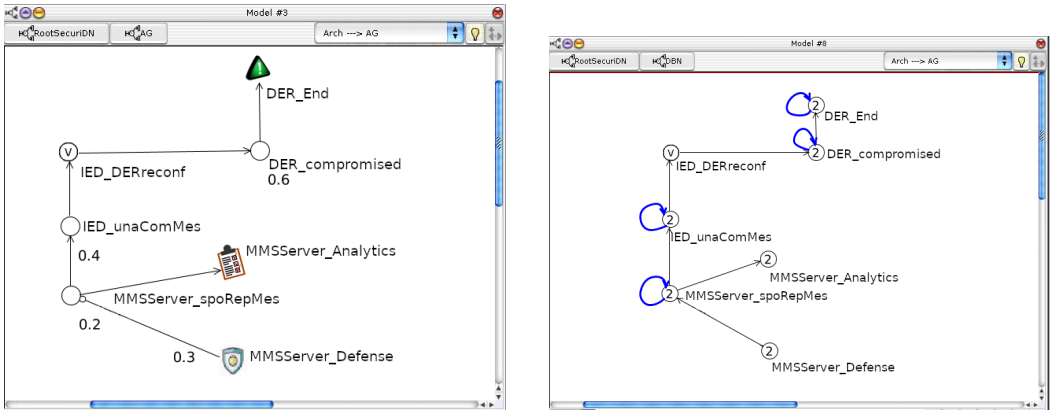
In the gAG in Fig. 3a the attack begins with the step *MMSServer_spoRepMes* (spoof reporting message) generating the analytic *MMSServer_Analytics* and mitigated by *MMSServer_Defense*. The possible success of *MMSServer_spoRepMes* enables the technique *IED_unaComMes* (unauthorised command message) whose success may compromise the DER (node *DER_compromised*) through the OR node *IED_DERreconf* (in the lAG of the DER, the OR node is connected to other nodes which have been deleted in the gAG during the reduction step because not reachable in this scenario). The arc connecting *DER_compromised* to *DER_End* indicates the end of the attack. The nodes *MMSServer_spoRepMes*, *MMSServer_Analytics*, and *MMSServer_Defense* come from the lAG of MMSServer (Fig. 2). The nodes *IED_unaComMes* and *IED_DERreconf* come from the lAG of IED. Finally, *DER_compromised* and *DER_End* come from the lAG of DER.

4.3.2. Dynamic Bayesian Network

The gAG is then converted into a DBN (Fig. 3b) with a compact representation (see Appendix A), where all associated state variables are binary. Each node in the gAG is translated to a DBN node, and each arc to a DBN arc. In this way, not only the dynamics of the whole attack is described, as in the gAG, but also the underlying stochastic attack process is modeled. Technique nodes are enriched with a self-loop temporal arc (colored in blue) to model the dependence of their state from the state at the previous time instant. A successfully executed technique influences the activation of the connected analytic to model the occurrence of an alarm. Through the CPT parameters we configure the rates of false positives and negatives of each analytic. On the other hand, a defense node connected to a technique node influences the activation of the latter, reducing, possibly to zero, its probability of success; this models the mitigation or inhibition effect of the defense measure.

Each node is also enriched with a CPT whose parameters can be manually or automatically set. In the former case using the GUI the user can inspect each node of the gAG and compile the corresponding CPT. In the latter, it depends on how the parameter values are automatically derived. Following the approach in [49] the user, for each technique-node of the gAG, must specify in the GUI an estimated mean Time to Compromise of the technique, and from all these values the conditional probabilities of each node are automatically computed. Alternatively, such parameters can be learned by measurements from a real system, or experimental testbed, or also extracted from synthetic simulation traces. In all these cases no further input is required to the SecuriDN user because the learning process will be performed by external tools.

The DBN model is then used as input to the detection module of the monitoring and detection platform. The module can return predictive or diagnostic results conditioned by the evidences (observations) about the events occurring in the monitored system, collected through the



(a) (b)

Figure 3: (a) Simplified global AG (b) Generated DBN

platform. Several examples of the types of results provided by the analysis of the DBN can be found in [1].

5. Conclusions and future works

SecuriDN is a promising tool to support security analysts in the early detection of adversarial activity and in the assessment of the cybersecurity posture of the system. It is designed with the electro-energetic system in mind.

The integration with the detection platform (see the introduction) has not been fully completed, yet. SecuriDN currently produces the description of the DBN, but in the near future it will be enriched to produce an Octave [50] script, to perform the DBN inferences; this script will be an input to a detection module in the platform. We are working on methods for automated parameterization of the DBN's CPTs, applying learning techniques by simulation traces.

Acknowledgements

The authors would like to thank Alberto Livio Beccaria for implementing *DNlib* library, and Marco Gribaudo for the development and the maintenance of *Draw-Net* over the years.

This work is original and has been supported by a joint collaboration between RSE S.p.A. and Università del Piemonte Orientale, financed by the Research Fund for the Italian Electrical System under the Three-Year Research Plan 2022-2024 (DM MITE n. 337, 15.09.2022), in compliance with the Decree of April 16th, 2018.

References

[1] D. Cerotti, D. Codetta-Raiteri, G. Dondossola, L. Egidi, G. Franceschinis, L. Portinale, R. Terruggia, A modular infrastructure for the validation of detection systems, in: H. Alh-

elou, N. Hatziargyriou, Z. Dongg (Eds.), Power System Cybersecurity, Springer, 2023, pp. 311–336. doi:10.1007/978-3-031-20360-2_13.

[2] J. Byres, M. Franz, D. Miller, The use of attack trees in assessing vulnerabilities in SCADA systems, in: International Infrastructure Survivability Workshop, Lisbon, 2004.

[3] P. C.-W. Ten, C.-C. Liu, M. Govindarasu, Vulnerability Assessment of Cybersecurity for SCADA Systems Using Attack Trees, in: IEEE Power Engineering Society General Meeting, 2007.

[4] P. C.-W. Ten, G. Manimaran, C. Liu, Cybersecurity for critical infrastructures: attack and defense modeling, IEEE Trans. on Systems, Man and Cybernetics, part A 40 (2010) 853–65.

[5] A. Roy, D. Kim, K. Trivedi, Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees, in: Int. Conf. on Dependable Systems and Networks, Boston, MA, 2012.

[6] D. Codetta-Raiteri, Generalized fault trees: from reliability to security, in: International Workshop on Quantitative Aspects in Security Assurance, London, UK, 2013.

[7] S. Bistarelli, F. Fioravanti, P. Peretti, Defense trees for economic evaluation of security investments, in: International Conference on Availability, Reliability and Security, IEEE Computer Society, 2006.

[8] B. Kordy, S. Mauw, S. Radomirović, P. Schweitzer, Foundations of attack–defense trees, Formal Aspects of Security and Trust (2011) 80–95.

[9] J. P. McDermott, Attack Net Penetration Testing, in: Workshop on New security paradigms, 2000.

[10] H. Henry, R. Layer, K. Snow, D. Zaret, Evaluating the risk of cyber attacks on SCADA systems via Petri net analysis with application to hazardous liquid loading operations, in: Conference on technologies for homeland security, IEEE, 2009, pp. 607–614.

[11] G. Helmer, J. Wong, M. Slagell, V. Honavar, L. Miller, Y. Wang, X. Wang, N. Stakhanova, Software fault tree and coloured Petri net–based specification, design and implementation of agent-based intrusion detection systems, Int. Journal of Information and Computer Security 1 (2007) 109–142.

[12] S. Pudar, G. Manimaran, C. Liu, PENET: A practical method and tool for integrated modeling of security attacks and countermeasures, Computers & Security 28 (2009) 754–771.

[13] M. Dacier, Y. Deswarte, Privilege graph: an extension to the typed access matrix model, in: Computer Security, Springer, 1994, pp. 319–334.

[14] O. Scheyner, Scenario Graphs and Attack Graphs, Ph.D. thesis, Carnegie Mellon University, 2004.

[15] E. LeMay, M. D. Ford, K. Keefe, W. H. Sanders, C. Muehrcke, Model-based Security Metrics Using ADversary VIew Security Evaluation (ADVISE), in: Int. Conf. on Quantitative Evaluation of Systems, 2011, pp. 191–200.

[16] M. J. Rausch, B. Feddersen, K. Keefe, W. H. Sanders, A comparison of different intrusion detection approaches in an advanced metering infrastructure network using ADVISE, in: Int. Conf. on Quantitative Evaluation of Systems, 2016, pp. 279–294.

[17] K. Keefe, B. Feddersen, W. H. Sanders, C. Muehrcke, D. Parks, A. W. Crapo, A. Gabaldon, R. Palla, Enterprise security metrics with the advise meta model formalism, in: International Conference on Emerging Security Information, Systems and Technologies, 2015.

[18] B. Kordy, L. Piètre-Cambacédès, P. Schweitzer, DAG-based attack and defense modeling: Don't miss the forest for the attack trees, Computer science review 13 (2014) 1–38.
[19] P. Xie, J. Li, X. Ou, P. Liu, R. Levy, Using Bayesian Networks for cyber-security analysis, in: Int. Conf. on Dependable Systems and Networks, 2010, pp. 211–220.
[20] S. Zhang, S. Song, A novel attack graph posterior inference model based on Bayesian network, Journal of Information Security 2 (2011) 8–27.
[21] A. S. M. Frigault, L. Wang, S. Jajodia, Measuring network security using dynamic Bayesian network, in: Workshop on Quality of protection, 2008, pp. 23–30.
[22] E. Mathias, J. Pontus, R. Lagerstrom, D.Gorton, J. Nydren, K. Shahzad, SecuriCAD by Foreseeti: A CAD Tool for Enterprise Cyber Security Management, in: Int. Workshop on Enterprise Distributed Object Computing, 2015, pp. 152–155.
[23] P. Johnson, R. Lagerström, M. Ekstedt, A meta language for threat modeling and attack simulations, in: Int. Conf. on Availability, Reliability and Security, 2018.
[24] P. Mueller, B. Yadegari, The stuxnet worm, Department of Computer Science, University of Arizona (2012). URL: https://www2.cs.arizona.edu/~collberg/Teaching/466-566/2012/Resources/presentations/topic9-final/report.pdf.
[25] Cybersecurity, I. S. Agency, CrashOverride Malware, https://www.cisa.gov/news-events/alerts/2017/06/12/crashoverride-malware, 2017. [Online; accessed 08/02/2024].
[26] M. Kezunovic, Y. Guan, C. Guo, M. Ghavami, The 21st century substation design: Vision of the future, in: IREP Bulk Power System Dynamics and Control Symposium, 2010.
[27] IEC TC 57 - Power systems management and associated information exchange, IEC 61850:2018 SER , 2018.
[28] M. M. Roomi, S. M. S. Hussain, D. Mashima, E.-C. Chang, T. S. Ustun, Analysis of false data injection attacks against automated control for parallel generators in iec 61850-based smart grid systems, IEEE Systems Journal 17 (2023) 4603–4614.
[29] IEC TC 57 - Power systems management and associated information exchange, IEC 62351:2024 SER, 2024.
[30] H. C. Tan, C. Cheh, B. Chen, D. Mashima, Tabulating cybersecurity solutions for substations: Towards pragmatic design and planning, in: IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia), 2019, pp. 1018–1023.
[31] D. C. Schmidt, Model driven engineering, *guest editor's introduction*, IEEE Computer, Special Issue on Model Driven Engineering (2006) 25–31.
[32] S. Gaonkar, K. Keefe, R. Lamprecht, E. Rozier, P. Kemper, W. H. Sanders, Performance and dependability modeling with Möbius, SIGMETRICS Performance Evaluation Review 36 (2009) 16–21.
[33] K. S. Trivedi, R. Sahner, SHARPE at the age of twenty two, SIGMETRICS Performance Evaluation Review 36 (2009) 52–57.
[34] M. Gribaudo, D. Codetta-Raiteri, G. Franceschinis, Draw-Net, a customizable multi-formalism, multi-solution tool for the quantitative evaluation of systems, in: International Conference on the Quantitative Evaluation of Systems, IEEE, 2005, pp. 257–258.
[35] D. Codetta-Raiteri, G. Franceschinis, M. Gribaudo, Defining formalisms and models in the Draw-Net Modelling System, in: International Workshop on Modelling of Objects, Components and Agents, 2006, pp. 123–144.
[36] D. Codetta-Raiteri, UML class diagrams supporting formalism definition in the Draw-Net

Modeling System, Technical Report TR-INF-2019-07-04-UNIPMN, Istituto di Informatica, Università del Piemonte Orientale, 2019. URL: http://www.di.unipmn.it/TechnicalReports/TR-INF-2019-07-04-UNIPMN.pdf.

[37] M. Beccuti, D. Codetta-Raiteri, G. Franceschinis, S. Haddad, A framework to design and solve Markov Decision Well-formed Net models, in: International Conference on Quantitative Evaluation of Systems, 2007, pp. 165–166.

[38] E. Naumovich, S. Bernardi, M. Gribaudo, ITPN-PerfBound: A performance bound tool for interval Time Petri Nets, in: International Conference on Tools and Algorithms for the Construction and Analysis of Systems, 2009, pp. 50–53.

[39] A. Bobbio, D. Codetta-Raiteri, S. Montani, L. Portinale, Reliability analysis of systems with dynamic dependencies, in: Bayesian Networks: A Practical Guide to Applications, John Wiley & Sons, 2008, pp. 225–238.

[40] D. Codetta-Raiteri, L. Portinale, A Petri net-based tool for the analysis of generalized continuous time Bayesian networks, in: Theory and Application of Multi-Formalism Modeling, IGI Global, 2013, pp. 118–143.

[41] L. Portinale, A. Bobbio, D. Codetta-Raiteri, S. Montani, Compiling dynamic fault trees into dynamic Bayesian nets for reliability analysis: the Radyban tool, in: Bayesian Modeling Applications Workshop, volume 268 of *CEUR Workshop Proceedings*, 2007.

[42] M. Beccuti, D. Codetta-Raiteri, G. Franceschinis, S. Haddad, Non deterministic Repairable Fault Trees for computing optimal repair strategy, in: International Conference on Performance Evaluation, Methodologies and Tools, 2008.

[43] J. Pearl, Probabilistic reasining in intelligent systems: networks of plausible inference, Morgan Kaufmann, USA, 1988.

[44] F. Jensen, T. Nielsen, Bayesian Networks and Decision Graphs (2nd ed.), Springer, 2007.

[45] K. Murphy, Dynamic bayesian networks: representation, inference and learning, Ph.D. thesis, University of California, Berkeley, 2002.

[46] C. Huang, A. Darwiche, Inference in belief networks: A procedural guide, International Journal of Approximate Reasoning 15 (1996) 225–263.

[47] K. Murphy, S. Russell, Rao-blackwellised particle filtering for dynamic Bayesian networks, in: Sequential Monte-Carlo Methods in Practice, Springer, 2001.

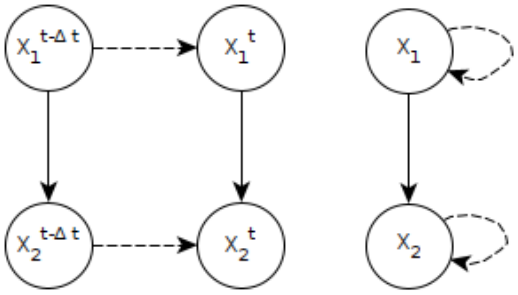
[48] X. Boyen, D. Koller, Tractable inference for complex stochastic processes. uncertainty in ai, in: International Conference on Uncertainty in Artificial Intelligence, 1998.

[49] D. Cerotti, D. Codetta-Raiteri, G. Dondossola, L. Egidi, G. Franceschinis, L. Portinale, R. Terruggia, A modular infrastructure for the validation of cyberattack detection systems, Technical Report TR-INF-2022-05-01-UNIPMN, Computer Science Institute, UPO, 2022. URL: https://www.di.unipmn.it/en/publications-en/technical-reports-en.html?pubid=567.

[50] J. W. Eaton, Octave, Accessed April 2022. https://www.gnu.org/software/octave/.

A. Appendix

Formal definition of BNs. A BN is a pair $N = \langle\langle V, E\rangle, P\rangle$ where $\langle V, E\rangle$ are the nodes and the edges of a *Directed Acyclic Graph* (DAG) respectively, and P is a probability distribution over V. Discrete random variables $V = \{X_1, X_2, \ldots X_n\}$ are assigned to the nodes, while each



(a) (b)

	X_1^t	$X_2^{t-\Delta t}$	X_2^t	pr.
1	0	0	0	1
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0.93
6	1	0	1	0.07
7	1	1	0	0
8	1	1	1	1

(c)

Figure 4: (a) The 2-TBN representation of a DBN; (b) its compact representation; (c) example of the CPT of node X_2.

edge $e \in E$ from node X to node Y represents a conditional dependency relationship between the variables represented by X and Y, where Y directly depends on X. This interpretation allows us to factorize the joint probability of the variables of the model, by considering only the conditional distribution of each variable with respect to their parent variables in the DAG: $P[X_1, X_2, \ldots, X_n] = \prod_{i=1}^{n} P[X_i|Parent(X_i)]$. Each local distribution can be described in a tabular form called *Conditional Probability Table* (CPT). Any kind of probabilistic query of the form $P(Q|e)$ can be computed, where Q is any set of unobserved variables and e is a configuration of a set of observed variables called the *evidence*.

Formal definition of DBNs. Given a set of time-dependent state variables $X_1 \ldots X_n$ and given a BN N defined on such variables, a DBN is essentially a replication of N over two time slices $t - \Delta t$ and t (being Δt the so called time discretization step), with the addition of a set of arcs representing the transition model. Let X_i^t denote the copy of variable X_i at time slice t, the transition model is defined through a distribution $P[X_i^t|X_i^{t-\Delta t}, Y^{t-\Delta t}, Y^t]$ where $Y^{t-\Delta t}$ is any set of variables at slice $t - \Delta t$ different from X_i (possibly the empty set), and Y^t is any set of variables at slice t different from X_i (possibly the empty set).

The dependencies of a certain node are quantified in terms of conditional probabilities and are stored in its CPT. The probability in every table entry has to be set according to the state of the parent nodes (possibly including the historical copy of the node). A DBN can be represented by explicitly drawing the two replicas of a 2-TBN, as shown in Fig. 4(a), or with a compact representation as in Fig. 4(b). In both cases two types of arcs are defined: *intra-slice* arcs indicate dependencies between nodes in the same time slice and are shown as continuous arrows; *inter-slice* arcs model dependencies between nodes in different slices and are depicted as dashed arrows. In the example X_2 depends on X_1 at the same time slice, and on its copy in the previous slice. Assuming that all nodes are associated with two-state variables, Fig. 4(c) shows an example of the CPT of X_2: each entry represents a specific state configuration of the parent nodes, and provide the probability that node X_2^t changes state to 0 or 1.