Go with the flow: Toward workflow-oriented security assessment

Conference Paper · December 2013

DOI: 10.1145/2535813.2535821

Binbin Chen
Advanced Digital Sciences Center

Zbigniew Kalbarczyk
University of Illinois, Urbana-Champaign

William Sanders
University of Illinois, Urbana-Champaign

Rui Tan
Nanyang Technological University

Go with the Flow: Toward Workflow-Oriented Security Assessment

Binbin Chen[*]
binbin.chen@adsc.com.sg

Zbigniew Kalbarczyk[†]
kalbarcz@illinois.edu

David M. Nicol[†]
dmnicol@illinois.edu

William H. Sanders[†]
whs@illinois.edu

Rui Tan[*]
tanrui@adsc.com.sg

William G. Temple[*]
william.t@adsc.com.sg

Nils Ole Tippenhauer[*]
nils.t@adsc.com.sg

An Hoa Vu[*]
anhoa.vu@adsc.com.sg

David K.Y. Yau[‡*]
david_yau@sutd.edu.sg

[*]Advanced Digital
Sciences Center
1 Fusionopolis Way
Singapore 138632

[†]University of Illinois
at Urbana-Champaign
1308 W. Main Street
Urbana, IL 61801

[‡]Singapore University
of Technology and Design
20 Dover Drive
Singapore 138682

ABSTRACT

In this paper we advocate the use of *workflow*—describing how a system provides its intended functionality—as a pillar of cybersecurity analysis and propose a holistic workflow-oriented assessment framework. While workflow models are currently used in the area of performance and reliability assessment, these approaches are designed neither to assess a system in the presence of an active attacker, nor to assess security aspects such as confidentiality. On the other hand, existing security assessment methods typically focus on modeling the active attacker (e.g., attack graphs), but many rely on restrictive models that are not readily applicable to complex (e.g., cyber-physical or cyber-human) systems.

By "going with the flow," our assessment framework can naturally adopt a holistic view of such systems, unifying information about system components, their properties, and possible attacks to argue about a security goal. The argument is expressed in a graph structure, based on inputs from several distinct classes that are integrated in a systematic manner. That rigorous structure allows our approach to provide quantitative assessment in an automated fashion (like reliability assessment tools and attack graphs), while maintaining a broad assessment scope. We demonstrate our security assessment process using the case of Advanced Metering Infrastructure in a smart power grid and obtain quantitative results for system availability and confidentiality.

NSPW'13, September 9–12, 2013, Banff, AB, Canada.
Copyright 2013 ACM 978-1-4503-2582-0/13/09 ...$15.00.
http://dx.doi.org/10.1145/2535813.2535821.

Categories and Subject Descriptors

K.6.5 [**MANAGEMENT OF COMPUTING AND INFORMATION SYSTEMS**]: Security and Protection

Keywords

Security Assessment; Argument Graph; Workflow; Security Model; Complex Systems

1. INTRODUCTION

Over the years, the security community produced much work addressing the problem of *cybersecurity assessment*[1], with different approaches varying in intent, scope, and human effort requirement [15, 20, 21, 28, 30]. For example, security assessment in industry typically focuses on demonstrating compliance with a policy or standard (e.g., Common Criteria). Similarly, more general work such as the Methodically Organized Argument Tree (MOAT) [13] and security assurance cases [3, 6, 11] intend to present a structured security argument demonstrating that a system is "acceptably secure." However, because these security arguments must be constructed manually, they do not scale well and may be derived or interpreted differently by different stakeholders. Other work, largely academic in origin, is intended to quantify the security of specific aspects of systems. Two prominent efforts are attack-graph-based security quantification [10, 26, 30, 38] and attack surface [20] methods, which are commonly used for security assessments of IT networks and software, respectively.

In recent years, security assessments of various complex systems have gained attention, e.g., in the context of voting processes [17, 27, 32] and smart power grids [7]. Different assessment methods have been proposed for specific scenarios, to model the involvement of physical processes and human actors. However, we perceive the need for a general security assessment framework that will be able to assess a wide range of scenarios, among them traditional cyber-systems, cyber-physical systems and cyber-human systems. Such an

[1] We will use *cybersecurity* and *security* interchangeably.

approach needs to be both flexible enough to model physical, cyber, and human interactions and formal enough to allow automated reasoning, which promotes scalability.

In this work, we propose a framework to address these challenges. Our approach is motivated by the following insights: (i) cybersecurity assessment should be carried out throughout the system design, implementation, and operation stages, (ii) diverse security-relevant information (evidence) is available throughout the system lifecycle, and may take different forms for different system components, and (iii) new tools are needed to aggregate this evidence into an overall assessment of the system's security. To this end, we propose a structured framework to integrate diverse information about the characteristics of a system, the functionality it provides, and possible threats, in a manner to reduce human effort through automation.

Our assessment framework is *workflow-oriented*: based on abstract descriptions of the actors and interactions in the system. That approach allows us to define which aspect of the system the assessment should consider, based on the activities and services it provides to/for stakeholders. A similar paradigm exists in the area of reliability analysis, e.g. [4], but such approaches are designed neither to assess the system in the presence of an active attacker, nor to assess security aspects such as confidentiality.

By "going with the flow," our approach naturally achieves a holistic perspective by using a workflow as the backbone to incorporate a variety of heterogeneous information about the system being assessed, which includes detailed system information, empirical data on system components, and possible attack models. Based on that diverse input, our framework derives a computable argument graph that captures the relevant interactions within the system and with possible attackers. The graph is then used to quantitatively evaluate attributes of the system such as confidentiality, integrity, and availability.

To summarize, our main contributions are the following:

- We identify the need for a holistic, yet rigorously structured approach to assess security aspects of complex systems and show the importance of considering workflows (describing how a system provides its intended functionality) to define the scope of the security assessment.

- We propose a novel workflow-oriented security assessment framework that: (i) integrates detailed system and attacker information to enable reasoning about security properties in a holistic manner, (ii) combines quantitative evidence to evaluate a security goal (e.g., "the workflow can be accomplished in spite of attacks or failures"), and (iii) supports a significant degree of automation.

- We present a case study and apply our approach to evaluate the availability and confidentiality of an example smart grid Advanced Metering Infrastructure.

This work is structured as follows: In Section 2, we argue that a workflow-oriented approach can improve security assessments of complex systems. In Section 3, we present a framework to support such a workflow-based security assessment. We then apply the proposed method to a smart grid case study in Section 4. In Section 5, we discuss our experimental implementation of the framework and highlight its automation potential. We compare our approach with related work in Section 6. We conclude the paper and discuss future work in Section 7.

2. MOTIVATION

In this section, we identify gaps in existing methods for security assessments, and summarize features required to bridge these gaps. We then propose a workflow-oriented assessment framework that provides those features.

2.1 Gaps in Cybersecurity Assessment

The performance and reliability community has identified and solved many challenges in modeling and understanding today's complex systems. Such assessments may include detailed *system models* as well as rigorous mathematical underpinnings. In recent years, some reliability assessment approaches have even been extended to consider high-level *workflow models* intended to represent the processes through which the system provides its intended functionality [23]. However, the theoretical gap between dependability and security is well-known [25]; this gap is mainly due to the absence of an *attacker model*, as well as less-developed methods for assessing information properties such as confidentiality and integrity.

Conversely, the security community has directed a significant effort toward modeling attackers and their interactions with an IT network or piece of software. Examples of such work include the software attack surface metric and its automatic derivation process [20] and the various methods for constructing attack graphs [30, 38] to model an attacker's path to a target host on a network. However, we observe that many such approaches are not readily applicable to complex systems, particularly because of the absence of *workflow models*, which help bridge the gap between IT infrastructure and other subsystems or actors that influence system operation.

Recent work has begun to address that gap., e.g., [2, 5, 12, 17, 27, 32]. For example, the *Human-Influenced Task-Oriented Process (HITOP)* modeling formalism uses workflow specifications to reveal how human actors influence critical system processes [5]. However, this formalism by itself does not include the attacker-specific information needed for a holistic security assessment. Similarly, workflow information has recently been considered in the context of voting process security [17, 27, 32] to enable the application of well-known IT security tools like attack trees to systems with human interactions. In contrast to such bespoke approaches, we recognize the need for a broad framework, that accepts input information in a range of formats and automatically derives a holistic security argument and quantitative assessment. We develop such a framework in the following sections, and defer further discussion of related work to Section 6

2.2 Hallmarks of a New Approach

We now identify several directions for improving the state-of-the-art in cybersecurity assessment, and highlight their importance with brief examples inspired by smart power grids. We provide more details on smart grids in Section 4.1.

Explicit consideration of workflow. While workflow information has been used in other security related areas (e.g., the design of specification-based intrusion detection [14] and provenance systems [24]), a formal workflow model is usually omitted from existing security assessment approaches. Nevertheless, such information can be critical to proper understanding of security risks. A workflow model describing the activities and services a system provides to/for stakeholders can serve as a natural backbone linking cyber and physical system elements as well as other information, like attacker models.

As an example, let us consider the security of Advanced Metering Infrastructure (AMI) in smart power grids. In recent years, people have become increasingly concerned about a potential attack that exploits the remote disconnect capability of smart meters to cause a large-scale blackout [1]. To model this, existing approaches like attack trees or the attack execution graphs used in [18], mainly focus on the analysis of the steps that an attacker takes in order to send the malicious commands to the meters. However, such an analysis may not be sufficient to capture the overall impact of that potential attack. If the workflow information about the utility's remote disconnect process is considered, one may show that surreptitiously sending a large number of disconnect commands does not necessarily lead to a large-scale blackout. In particular, when the process of switching meters on and off is implemented with a random delay on the order of hours [34, 35], that delay would cause some households to be affected earlier than others. With some of those affected customers promptly reporting their loss of power to the utility's customer service department, the utility company will likely be able to detect and react to the attack, hence prevent the remaining customers from actually losing their power. That workflow, which involves human actors, allows the utility to discover (and potentially stop) an attack before it is fully realized, reducing the risk of a widespread blackout.

Holistic perspective. An assessment should adopt a holistic view, that considers detailed system information, such as connectivity and device settings, along with sophisticated attacker models and the workflows that describe the system's core functionality and interactions with users and its environment. For an assessment to be truly holistic, all inputs must be integrated at a low level and unified into one common security argument.

To continue the AMI example, consider a group of smart meters that uses a wireless network to communicate with a local aggregator. One important design decision for the utility is whether or not to support dynamic routing in their meter network. Analysis using traditional dependability tools (without an attacker model) may favor dynamic routing to increase reliability. However, in the presence of an attacker, dynamic routing may lead to a severe loss of availability (e.g., because of a black hole attack [8]), and may also lead to confidentiality consequences if a household's meter readings are routed through a compromised device. On the other hand, attacker-centric methods that do not consider the internal operations of the system (the workflows) may undervalue performance gains in normal operating conditions. To properly assess the trade-off one needs to consider high-level user requirements, as well as the way different components interact, which is embedded in the workflow (see also Section 4.3).

Structured integration framework. A holistic security assessment must have precisely defined input classes and clear semantics for the security argument. Meeting this requirement is essential for removing ambiguity and promoting continuous refinement and re-evaluation of the security assessment throughout the system lifecycle. Without such structure, different experts or decision-makers may construct or interpret the security assessment differently.

For example, the security assurance case method [6] is general enough to capture workflow within its goals, claims, arguments, and evidence. It may seem reasonable, then, to use security assurance cases to convince stakeholders that "The smart meter remote disconnect command cannot be abused to cause a blackout." However, since that formalism does not provide a precise step-by-step framework *to build the arguments* and/or *to evaluate them afterward*, several utilities in the same region could reach different conclusions about the security of their systems, even if they use identical business processes (workflow of the disconnect command) and the same type of smart meters.

Support for automation. An important strength of many IT security assessment methods is that they can automatically construct some security argument or derive a certain quantitative metric by analyzing the source code of a program [20] or the topology of a network [30, 38]. The size and heterogeneity of complex systems necessitates similar automation, yet existing approaches do not attempt to achieve this. We identify the need for such automation, and propose that an assessment process with the above features—explicit consideration of workflow, a holistic perspective, and a structured integration framework—can support automated argument derivation and evaluation.

The AMI examples discussed previously (dynamic routing and remote disconnect) illustrate the security implications of physical design choices as well as business processes. While AMI systems are undoubtedly complex (and part of an even larger ecosystem of devices and services in a smart grid), we can unify relevant security information in an automated way by adopting a workflow-oriented approach. We investigate the issue practically in Section 5, using publicly available AMI meter-reading workflow models.

2.3 Going with the Flow

We propose a new framework for reasoning about the security of complex systems. Our approach incorporates five input classes: *security goal*, *workflow description*, *system description*, *attacker model*, and *evidence*.

We envision an assessment process that begins at a high level with a workflow model, and becomes progressively enriched with system-specific information and attacker-specific information. A natural use case for this framework is design-time security analysis, where a system architect with access to detailed documentation (and potentially security experts) can evaluate design alternatives and capture underlying security assumptions. In fact, the structure of our workflow-oriented assessment framework parallels the system design and development process. Initial design typically begins with a set of functional requirements expressed as workflow diagrams, with no specific implementation information available. As the design progresses, a network topology becomes available, allowing the generic workflows to be instantiated for specific devices, software, protocols, and network topology. Once the system is described in sufficient detail, different attacker models may be used to understand weaknesses and iteratively improve the design.

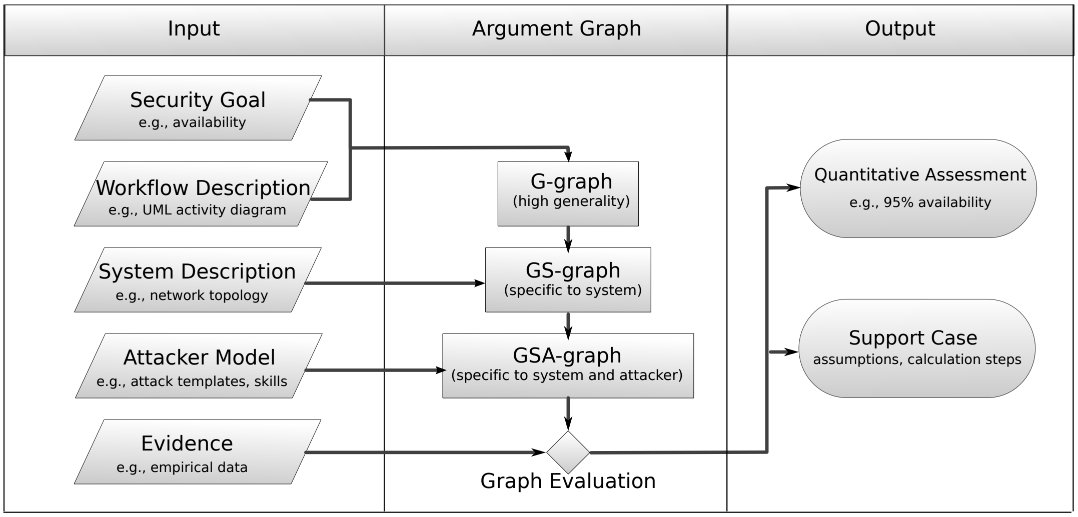


Figure 1: Proposed framework to automatically generate and evaluate an argument graph.

In addition to this design-time analysis, our assessment framework can also be used when the system is reconfigured, to verify that initially assessed security properties still hold. After the design process, the security argument can thus form part of the system specification. If real-time evidence is provided, one could also envision the use of our framework for on-line security monitoring of a running system.

Regardless of the specific application, we believe the modular and structured nature of our security assessment framework will offer several benefits to individual users and organizations. For example, different stakeholders often represent their unique (and perhaps incomplete) system knowledge in different formats: a system architect may use tools like UML activity diagrams to express system workflows, while security analysts may represent attack scenarios using attack trees. Rather than asking these stakeholders to learn a new modeling formalism, our framework, and the prototype software tool described in Section 5, allows different users to input this information in its original form. This reduces modeling effort while providing these users with a global view of system security that may have been previously unavailable.

3. WORKFLOW-ORIENTED SECURITY ASSESSMENT

To derive arguments about the security of a system, the assessment process in our framework takes a top-down approach (Fig. 1). Initially, the process starts with the *workflow* under consideration and the *security goal.* Based on the workflow and goal, the Goal graph (*G-graph*) is constructed, which decomposes the security goal into abstract intermediate goals. As the next step, a security analyst uses concrete information about the *system*, such as the network topology and configuration of individual devices, to instantiate the abstract intermediate goals in the context of the system under consideration. This results in the Goal and System graph (*GS-graph*). The leaf nodes of the GS-graph correspond to basic requirements, e.g., for availability of components. The assessment process then takes the *attacker* into consideration, and uses information about attacker strategies and possible entry points to extend the GS-graph to the Goal, System and Attacker graph (*GSA-graph*). As a last step, the assessment process aggregates *evidence* about leaf requirements and attaches it to the graph. That final graph is then evaluated by the process to compute the assessment result.

3.1 Inputs

Security goal. A security assessment requires clearly defined security requirements. In the proposed framework, the security goal can be a combination of security aspects such as confidentiality, integrity, and availability, and is usually defined from a system perspective (instead of an attacker perspective). Thus, an example goal could be "Workflow executes successfully and preserves the confidentiality of all exchanged information." Note that a goal can apply to multiple workflows.

Workflow description. The workflow description defines the specific task(s) that should be assessed with respect to the defined goal. This input describes *how the system provides value*, and should identify necessary actors (e.g., abstract devices or humans) as well as the information they exchange. To facilitate use of the framework, the workflow input should accept a range of formats, e.g., UML activity diagrams or HITOP models [5].

System description. While the workflow input describes abstract actors and processes, the system description contains concrete information about the *specific system under evaluation.* This information covers as many aspects of the system as possible, and is needed to accurately characterize the system. This input could include network topology, device specifications, or software profiles.

Attacker model. The attacker model input contains information about potential adversaries, e.g., the attacker's access (specific locations, remote access), privilege (root, authenticated, unauthenticated), and skill. The attacker models can be organized in *attacker profiles* to allow for easy switches between different attacker configurations. This information can be used to obtain attacker-specific *attack tem-*

plates that detail a sequence of individual attacker actions an attacker could use to achieve his objectives.

Evidence. While the above inputs are used to construct a security argument, evidence is used to evaluate the system with respect to the argument's goal. Since different components of a system may be assessed by different techniques (such as expert opinion, discrete event simulation, or empirical data), all evidence needs to be translated into quantitative values.

3.2 Argument Graphs

Our framework aims to automatically aggregate the diverse input information described above. In the center of our framework is the construction of *argument graphs*, which capture the logical relations of these diverse inputs in a holistic and structured manner. Our argument graphs are constructed in several stages based on the workflow under consideration, as illustrated in Fig. 2. The process starts with the G-graph and results in the final GSA-graph. After the GSA-graph has been constructed, it can be used for a quantitative security assessment of the system.

The argument graphs consist of nodes and directed edges. The edges carry information that is used in the graph-driven evaluation/reasoning process. We envision that the graph structure supports assessments that use different methods to aggregate information delivered to each node (non-leaf) by the incoming edges (we discuss a concrete example in Section 3.4). We introduce two basic node types—action and property nodes—describing system behavior, along with two equivalent node types describing *attacker* actions and properties. In either case, action nodes correspond to events that change the state of the system while property nodes describe characteristics that are not directly associated with state changes.

G-graph. The workflow description and the goal are combined in the framework to derive the first high-level argument graph, which we call a *G-graph.* The G-graph provides a mapping of the goals to the individual abstract components involved in the workflow, while preserving the inherent sequential structure of the workflow under consideration. The G-graph is a directed graph with three different kinds of nodes: *action nodes*, *property nodes*, and *start/goal* nodes (Fig. 2). The node types differ in the data they represent, and how they can be connected to other nodes. The start/goal nodes are the simplest: the start node denotes the start of the workflow under consideration, and connects directly to the first action in the workflow. The goal node denotes the completion of the workflow, and aggregates all properties of the workflow execution. Both nodes together form the set $\mathbb{G}$.

An action node $A_i \in \mathbb{A}$ relates to actions taken in the target workflow. For example, an action node could describe a field device transmitting a measurement, and is used to aggregate the associated properties. An action node is denoted by an ellipse in the graphs. Action nodes connect to other action nodes or start/goal nodes using *workflow step* edges denoted by a thick solid arrow $N_i \rightarrow N_j$ with $N_i \in \mathbb{A}$ and $N_j \in \mathbb{A} \cup \mathbb{G}$.

A property node $P_i \in \mathbb{P}$ aggregates properties. For example, this property could be the availability of a certain network link. Property nodes connect to each other or action nodes using a *composition edge* in the graph: $N_i \dashrightarrow N_j$ with $N_i \in \mathbb{P}$ and $N_j \in \mathbb{P} \cup \mathbb{A}$.

GS-graph. Based on the previously defined G-graph and the system description, the framework derives the *GS-graph.* This more detailed representation maps the abstract actors from the workflow description to actual hardware, including information about component redundancy and the nature of the communication links, such as information about network traffic passing through a firewall, or a multi-hop path in an ad-hoc network.

As the GS-graph builds on the G-graph, it can contain all node and edge types of the G-graph. In addition, the GS-graph can contain *leaf property nodes*, which form the set $\mathbb{L}$. Leaf property nodes are very similar to normal property nodes, but cannot be decomposed further. Leaf property nodes define requirements for evidence; the evidence is going to be attached to these nodes in a later step. Leaf nodes use the same composition edges to connect to other property nodes or action nodes: $N_i \dashrightarrow N_j$ with $N_i \in \mathbb{L}$ and $N_j \in \mathbb{P} \cup \mathbb{A}$.

GSA-graph. Based on the GS-graph and the attacker model, the assessment process derives the *GSA-graph.* The GSA-graph integrates the knowledge about the attacker into the GS-graph. For example, weak points identified in the GS-graph are matched to attack templates to model the set of possible actions for the attacker.

The GSA-graph introduces two new types of nodes for the graph: *attacker action nodes*, and *attacker property nodes.* Attacker action nodes denote activities taken by the attacker in the system, and can also aggregate properties. For example, such an action could be "execute remote exploit on Meter 1". Attacker action nodes form the set $\mathbb{M}$ and connect to other attacker action nodes or to *attack entry points* in leaf property nodes, using *attack step edges* (thin and red): $N_i \rightarrow N_j$ with $N_i \in \mathbb{M}$ and $N_j \in \mathbb{M} \cup \mathbb{L}$.

Attacker property nodes from the set $\mathbb{S}$ denote individual properties (e.g., skills) of the attacker. For example, the attacker could have the skill to run a remote exploit on certain meters. These properties are defined in the attacker model, and attach to attack action node, represented by thin red attack step edges: $N_i \rightarrow N_j$ with $N_i \in \mathbb{S}$ and $N_j \in \mathbb{M} \cup \mathbb{L}$.

Argument graph evaluation. While the GSA-graph already captures the logical correlations between system components and the attacker, actual data such as attack success probabilities and statistics about device availability are still missing. Such data are provided to the assessment process as evidence input. Based on that input and the GSA-graph, the process can automatically evaluate the provided evidence (explained in more detail in Section 3.4).

3.3 Outputs

After the framework aggregates input information, the evaluation of that graph yields two results: a quantitative assessment and an associated support case. These outputs are mainly intended to help stakeholders compare the relative merits of different system architectures, and to gain a better understanding of factors influencing system security—and less to classify a system fully (in)secure.

Quantitative assessment. The quantitative assessment allows the evaluating party to understand how the system and its functional workflows perform with respect to the security goal. It is mainly intended to be compared with other results produced by this framework, rather than with outputs from other security evaluation tools. The nature of this quantitative assessment depends on the provided evi-

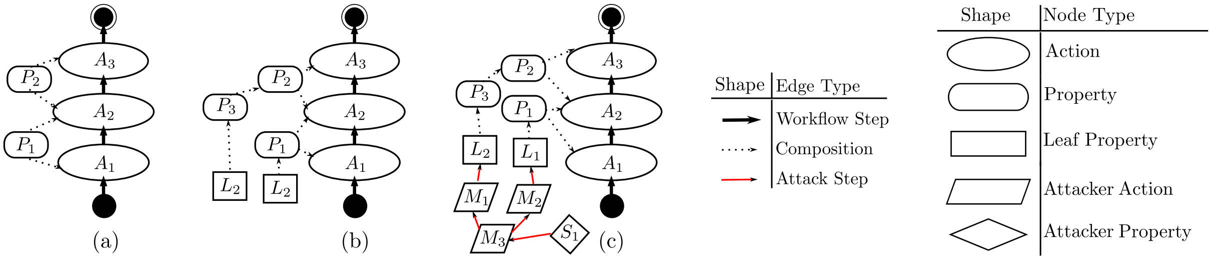


Figure 2: Argument graph construction: From G-graph (a) to the GS-graph (b) and finally GSA-Graph (c).

dence and the combination rules used to perform the aggregation. For example, if the estimated failure probabilities for components and network links are provided as evidence, our assessment process could provide an overall likelihood of successful workflow completion in the presence of various attackers.

Support case. Along with the quantitative assessment, the evaluation process provides a concise summary of the input conditions, calculation steps, and assumptions to engender confidence in the result. This support case can help the evaluator understand where problems lie in the current design, and assist with deciding which features should be strengthened first. For example, if the assessment result indicates that a workflow will complete successfully with 95% probability using a certain attacker model, the support case will provide the root cause analysis by presenting the different reasoning paths through the argument graph that account for the 5% probability of failure, as well as their relative importance.

3.4 Aggregation using Boolean Equations

So far, we have not specified how nodes aggregate information in the graphs, and what information is carried by edges. Evidence information can be aggregated in different ways; in the following, we discuss an approach that uses Boolean algebra to capture the logical relationships between evidence and the security goal (it is similar to techniques used in fault tree analysis [37]). In the assessment procedure, we model statistically independent information sources based on Boolean random variables. If such a Boolean random variable is True, then the connected event has occurred, or the connected property holds. The Boolean random variable A is True with a certain probability $P(A)$, derived from evidence in our case. Boolean random variables are put into relation with each other by equations using Boolean algebra. The assessment process ultimately aggregates a Boolean equation at the goal node. To obtain the final assessment result, that Boolean equation is then evaluated using numerical values obtained from the evidence.

Boolean equations. In our argument graphs, information is carried by edges between the nodes. If Boolean algebra is used for the assessment, the information is the *aggregated Boolean equation* (ABE) of the edge source. For any node, all outgoing edges carry the same information (the ABE of this node). Each node computes its ABE based on its *intrinsic Boolean equation* (IBE). The IBE operates on variables corresponding to the incoming edges. To obtain the ABE of a node, the process inserts all incoming ABEs into the node's IBE according to their placeholder variables. As a result, the ABE of that node is obtained and associated with all outgoing edges. Each node's IBE is defined during the graph construction, while the ABEs are computed and propagated during the graph evaluation phase. Once all incoming edges of the goal node have been populated, the ABE of the goal node can be constructed.

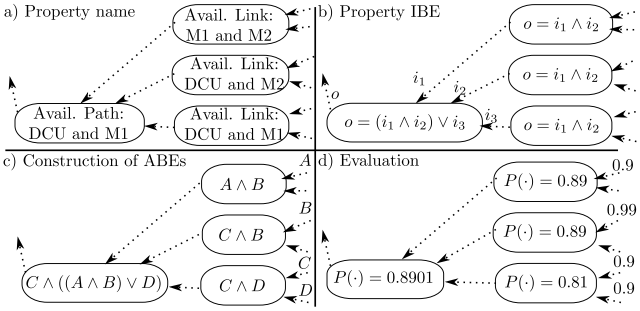


Figure 3: Example derivation of quantitative values based on IBEs and ABEs.

Leaf nodes without any incoming nodes must be initialized to some ABE to bootstrap the aggregation process. We do so by associating each leaf node with a Boolean random variable.

Graph construction. To evaluate the argument graph, each node's ABE must be derived based on the incoming ABEs and the node IBEs. After all ABEs have been computed, they can be evaluated to combine the quantitative data (from the evidence), which provides the central security argument. Shared Boolean random variables in the final ABE have to be considered (for example by conditioning in the evaluation process).

Example derivation of ABEs and result. In Fig. 3, we give an example of the process in more detail. In the upper left side, a small part of the GSA-graph is shown, with labels denoting the properties of the nodes. In this example, we consider two alternative network paths between the DCU and M1, similar to the detailed example in Section 4. One network path is a single-hop link, while the other one has an intermediate node. As a result, the abstract path is available if either the single-hop link or the alternative two links are available. The upper right side shows the intrinsic Boolean equations of the nodes, which get aggregated to the ABEs in the lower left side. Then, based on known probabilities assigned to underlying Boolean random variables, the quantitative values for the properties can be evaluated in the fourth step. For the numerical results, we consider the probability of path availability, and assume the following probabilities for the properties: $P(A) = P(C) = P(D) = 0.9$,

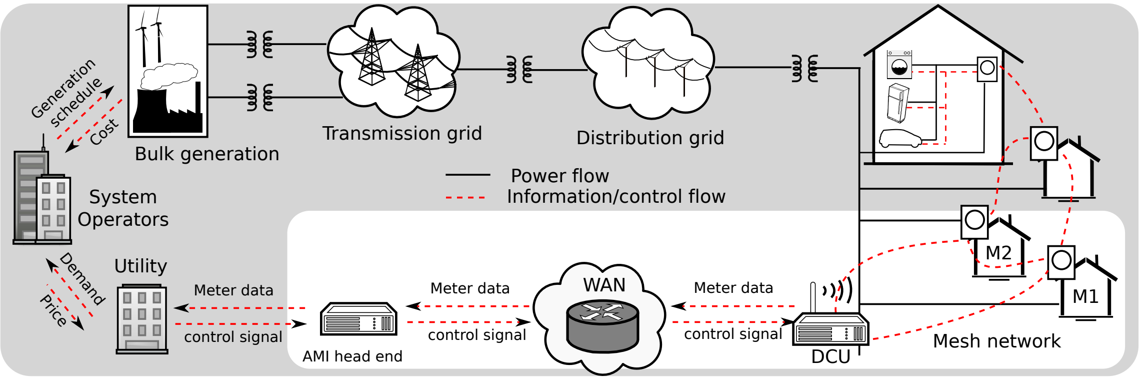


Figure 4: Power and data flow in a simplified smart grid system. Our case study considers its AMI component (the white region).

and $P(B) = 0.99$. In the figure, we use $P(\cdot)$ to denote the probability of the node's ABE being true. As the resulting ABE for the availability of the path between the DCU and M1 does not contain terms with shared Boolean random variables, the evaluation is straightforward.

4. SMART GRID CASE STUDY

In this section, we explain an assessment process based on ABEs and IBEs in more detail, using an example workflow from the smart grid domain as an illustration. We start by briefly introducing smart grid systems to provide context.

4.1 Smart Grid Overview

The term *smart grid* refers to a modernized power grid, which is characterized by broad measurement and control capability, primarily at the network edges. One example of a smart grid system deployed at the distribution level is *Advanced Metering Infrastructure* (AMI). The architecture of a typical AMI system is shown in the context of a power network in Fig. 4. The system consists of numerous *smart meters*, which provide the utility with fine-grained household energy consumption data and the ability to transmit dynamic price signals (e.g., time-of-use rates or real-time pricing) to influence consumption behavior.

All information sent to and from the meters is routed through *data concentrator units* (DCUs), which are located in individual neighborhoods. These neighborhood area networks (NANs) are typically implemented using powerline communication or wireless mesh networks. The head end is connected to the utility's corporate network, enabling access to other services which are omitted here, such as meter data archives or customer information systems.

The large scale of many AMI deployments, on the order of 500,000 meters or more (see Table 3.1 in [36]), and the likely reliance on wireless networks present complex information security and privacy issues. AMI security has been much discussed in recent years, with research focusing on privacy [31], energy theft [22], misuse of remote service disconnection [1], denial of service, or some combination thereof [7].

4.2 Detailed Assessment Process

In the following sections, we describe the steps of the assessment in detail, and also derive some example graphs. We will discuss automation of this process in detail in Section 5.

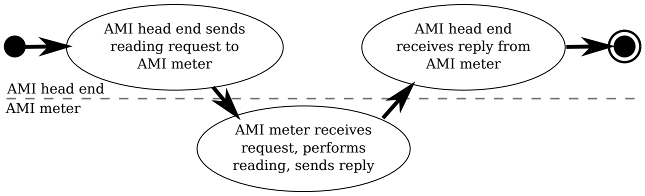

AMI head end sends reading request to AMI meter
AMI head end receives reply from AMI meter
AMI head end
AMI meter
AMI meter receives request, performs reading, sends reply

Figure 5: Smart meter on-demand reading workflow.

Goal and workflow model. The assessment process starts with the definition of the assessment goal. In our example case study, the security goal is to assess the probability that the workflow will finish successfully in the presence of an attacker (availability), and the probability that the attacker will obtain the meter reading (confidentiality). In our example, a head end unit at a utility initiates an on-demand reading of a smart meter at the customer site (see Fig. 5).

As the next step, our assessment process can automatically combine the workflow and goal to construct the G-graph, as shown in Fig. 6. In the G-graph, the workflow from the workflow description input has been extended to explicitly model exchanged messages. These messages are inferred from the workflow description, whenever two sequential workflow states are not associated with the same abstract actor. In addition, the G-graph contains nodes to explicitly model the actors: the AMI head end and the AMI meter to be read. The workflow also shows that an abstract path between meter and head end must be available.

System model. In the next step, the properties identified in the G-graph are further decomposed. This decomposition takes the system description into account. In our case study, the system description includes a network topology (Fig. 4), and information on the individual network links (e.g., the encryption used). In this network, the head end communicates with a local data concentrator unit (DCU), which itself forwards communication to two individual meters (M1 and M2). The meters and DCU form a local mesh network that uses preshared symmetric keys to encrypt all traffic. All traffic from the DCU to the head end is encrypted with a key shared only between the DCU and the head end.

The system description allows us to map the abstract actors from the workflow description to physical devices: we

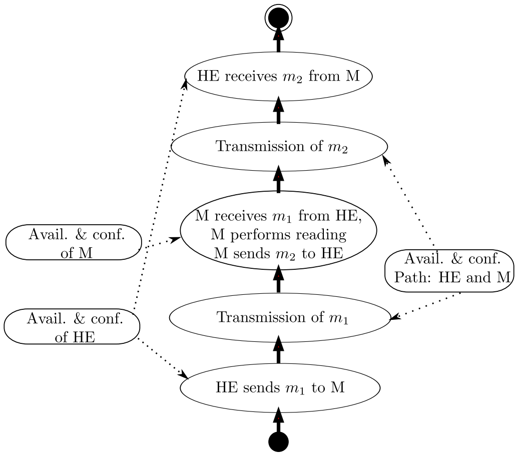


Figure 6: G-Graph for the example workflow. Here, M is the meter to be read, HE is the head end, and m_1 and m_2 are messages (request/response) inferred by the process.

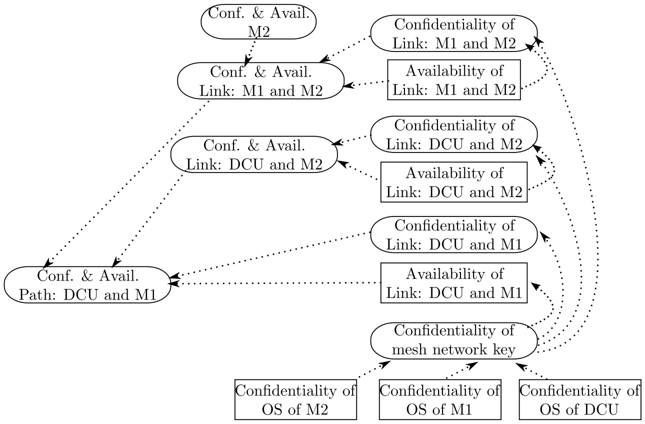

Conf. & Avail. M2
Confidentiality of Link: M1 and M2
Conf. & Avail. Link: M1 and M2
Availability of Link: M1 and M2
Confidentiality of Link: DCU and M2
Conf. & Avail. Link: DCU and M2
Availability of Link: DCU and M2
Confidentiality of Link: DCU and M1
Conf. & Avail. Path: DCU and M1
Availability of Link: DCU and M1
Confidentiality of mesh network key
Confidentiality of OS of M2
Confidentiality of OS of M1
Confidentiality of OS of DCU

Figure 7: A branch in the GS-graph for our example workflow, showcasing the path decomposition.

map the AMI meter from the workflow description to M1 in the network topology, and use the head end as defined in the topology. Based on the inference of a message exchange from the workflow description, the process discovers that the workflow relies on the availability of an abstract network path between the head end and M1. Using the topology, the process traces this abstract path through the DCU and determines that the availability of the DCU is also critical for the workflow completion, although the DCU was not explicitly mentioned in the workflow description. The process also uses the mesh network between the meters and the DCU to model two alternative routes between the DCU and M1: (i) a direct link, and (ii) an indirect path via M2, which would forward the messages.

In this example, actors are decomposed to the following properties: availability on the logical layer, availability on the physical layer, and confidentiality on the logical layer. Single-hop network links are decomposed into availability on the physical layer (e.g., noise, collisions) and confidentiality (e.g., eavesdropping). To reduce the complexity of the example, we do not decompose most of the properties further, but consider them as leaf property nodes to which evidence is attached. The confidentiality of links in the mesh network is decomposed into the confidentiality of the shared key, and the inverse of the physical layer availability of the link (Fig. 7).

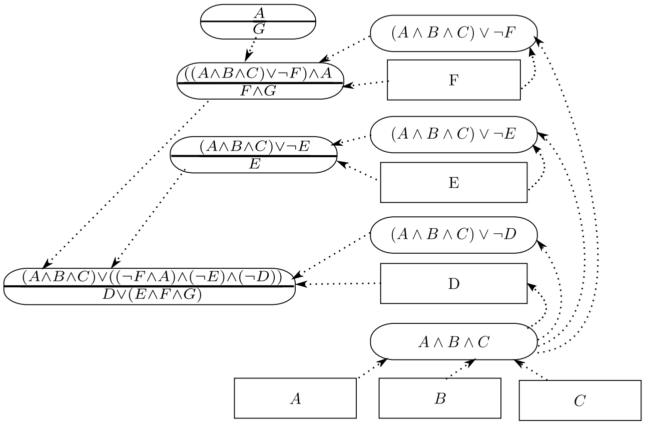
Attacker model. The attacker model defines template graphs of attacker actions that eventually connect to leaf property nodes in the GS-graph. For example, an attack template can consist of two attacker action nodes, connected with an attack step edge. In our example, one of the attack points is `OS integrity`. A matching attack template would consist of attacker actions: `Gaining Access to Location` and `Using local exploit`. The attacker can only consider executing the second action (local exploit) when the first action (gaining access) has been executed. For both actions, the attacker needs to have the required skills, which are defined separately for the attacker action template. Typical multi-stage attacks can also be collected and re-used in different assessments, independently of the attacker's skills.

In our example, the attacker has the skill to compromise the OS of any device, if he gets physical access to that device. The attacker is assumed to be able to gain access to the location of M2, but he not able to gain access to the locations of M1, the DCU, or the head end. Based on this attacker model input, the GSA-graph can be constructed–for this case study, the GSA-graph has 60 nodes and 82 edges. Due to this size, we cannot give a detailed view of the whole graph here. Instead, we provide an overview of the GSA-graph for this case study in Section 5, where we use our prototype tool to automatically generate it (Fig. 10).

Evidence. For our example workflow, the following evidence is needed to provide an assessment of the workflow's availability: (i) (statistical) data about the availability of the OS functionality for M1, the DCU, M2, and the head end, and (ii) (statistical) data about the availability of the hardware functionality for M1, the DCU, M2, and the head end. That evidence does not consider the attacker; it is similar to the statistical data on device failures used in reliability analysis. In our example, we assume a mean-time-to-fail of 5 years for all involved devices (hardware errors), a nonzero chance of software errors (1%) in the meter devices, and no software errors in the DCU and head end. For the links, we assume no message loss on the link between the DCU and head end, and a 10% loss rate for wireless links in the mesh network.

Computation of quantitative results. In our case study, we construct and evaluate the argument graph based on the Boolean random variable calculus described in Section 3.4. In Fig. 8, we show an example of how the ABEs are constructed based on the graphs (without showing the attacker). We use a horizontal bar to differentiate two separate properties: the upper ABE is related to confidentiality, and the lower one to availability.

The assessment has two quantitative results: an *availability score* (AS) and a *confidentiality score* (CS). We now start with assessment results in the absence of an attacker. In this setting, the availability of the meter-reading workflow is determined by general link quality and device availability. The quantitative result is computed by combining the probabilities for system operation and attacks as provided from the evidence. First, the ABE for each node in the GSA graph is computed (see Section 3.4 and Fig. 8). The combination of



A
G
$((A\wedge B\wedge C)\vee\neg F)\wedge A$
$F\wedge G$
$(A\wedge B\wedge C)\vee\neg E$
E
$(A\wedge B\wedge C)\vee((\neg F\wedge A)\wedge(\neg E)\wedge(\neg D))$
$D\vee(E\wedge F\wedge G)$
$(A \wedge B \wedge C) \vee \neg F$
F
$(A \wedge B \wedge C) \vee \neg E$
E
$(A \wedge B \wedge C) \vee \neg D$
D
$A \wedge B \wedge C$
A
B
C

Figure 8: The branch from Fig. 7 with annotated ABEs instead of labels. We separate confidentiality and availability properties with a horizontal bar.

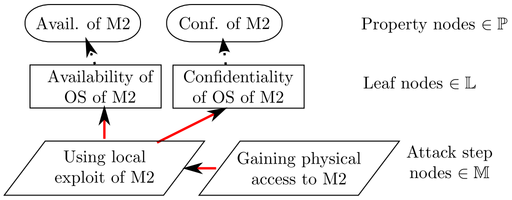


Figure 9: The trace of an attack in our example.

evidence then uses the ABE at the goal node to detect and resolve statistical dependence of evidence.

To illustrate the computation of the availability and confidentiality score in more detail, we continue with the example ABE as shown in Fig. 8. Here, the availability ABE of the path between the DCU and M1 is given as $D \vee (E \wedge F \wedge G)$. Using the evidence as detailed above, we arrive at an overall AS of 0.981 for that path. For the whole workflow, the AS is 0.97. As the communication between M1 and the DCU is encrypted, and the communication between the DCU and the head end is encrypted as well, confidentiality of the meter data is preserved ($CS = 1.0$).

When the attacker is considered to be able to compromise M2 locally, the availability score of the system is impacted, because the alternative path from M1 to the DCU via M2 is controlled by the attacker. A trace of this attack is shown in Fig. 9. If the probability of this attack (that makes M2 unavailable) is modeled as a Boolean random variable Z (which is statistically independent of other variables), then the availability of the path between the DCU and M1 is now given by the following ABE: $D \vee (E \wedge F \wedge (G \wedge \neg Z))$. The impact of such an attack can vary, depending on the attacker's goals. The attacker could either impact availability by making the path unavailable (with $Z = 1.0 \rightarrow AS = 0.891, CS = 1.0$), or forward the traffic while eavesdropping on all received data content, which would reduce the confidentiality score ($AS = 0.97$ and $CS = 0.1$). The attacker could also opt not to forward and eavesdrop, which would lower both the availability and confidentiality scores ($AS = 0.891, CS = 0.1$).

4.3 Discussion of Results

Our previous numerical examples demonstrate the importance of taking a holistic approach. In particular, different attacker models (e.g., whether an attacker makes his or her controlled path unavailable) could lead to different security assessment results. Thus, in order to help assess the impact of alternative design choices on the overall security of the system, one needs to consider the attacker models together with the system description. To illustrate that further, consider the design choice regarding whether one should enable dynamic routing in a wireless mesh network of meters. If dynamic routing is enabled, there is a certain risk that an attacker can launch a black-hole routing attack, in which a compromised meter announces a good route in order to attract and drop packets from other meters. If the attacker model suggests a high risk of such an attack, our security assessment result would show a dramatic decrease of the AS value because of dynamic routing. In this particular context, our holistic approach helps reveal a somewhat counter-intuitive insight that dynamic routing (to improve availability) should be disabled to ensure high availability.

Our meter-reading example also demonstrates the importance of making the assessment workflow-oriented. In practice, the simple meter-reading workflow serves as a building block for a high-level automatic monthly electricity billing workflow. Our example suggests that one needs the billing workflow to provide the necessary context for conducting the security assessment. In particular, the billing workflow may show that continuous real-time measurements are not needed: the security goal could be to obtain monthly readings. The billing workflow could also specify additional mechanisms to mitigate temporary reading failures (e.g., automatic retries). Thus, our workflow-oriented assessment framework allows one to better compare the risk and the gain associated with dynamic routing.

5. INITIAL PROTOTYPE

In the previous sections, we have described our framework on an abstract level (Section 3) and provided a detailed example of how the framework assesses a basic workflow (Section 4). While the example was derived by hand to simplify the figures and show internal values, one of our main goals is to provide a tool that allows the user to automatically assess the security of a system based on a set of inputs. To that end, we are currently working on an implementation of such an assessment tool. In particular, we use this prototype to experiment with automatic processing of user input, visualization with automatically generated graphs, and automated computation of the quantitative results.

The current version of the tool is written in Python and uses several external libraries to facilitate input parsing and graph generation. In particular, the tool reads workflow description inputs defined in XMI, an XML dialect for UML models. We currently use workflows in XMI format from [33] as test cases. The collection of workflows handles typical cases related to AMI meter readings. Our tool does not yet fully support branching and similar operations in the workflows. In addition to workflow descriptions, the tool processes system description inputs consisting of network topologies and mappings between abstract actors and hardware devices. The network topologies are parsed from XML files generated by the CSET tool [9].

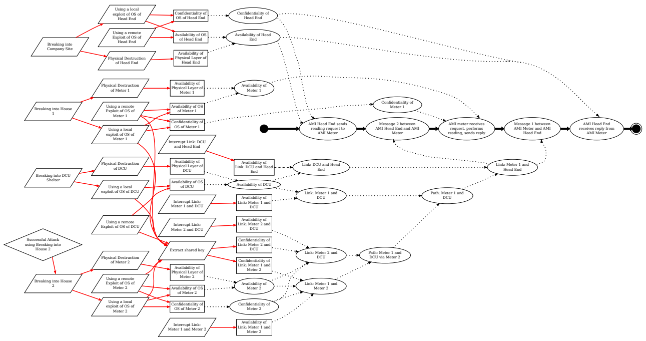

Breaking into Company Site
Using a local exploit of OS of Head End
Using a remote Exploit of OS of Head End
Physical Destruction of Head End
Confidentiality of OS of Head End
Availability of OS of Head End
Availability of Physical Layer of Head End
Confidentiality of Head End
Availability of Head End
Breaking into House 1
Physical Destruction of Meter 1
Using a remote Exploit of OS of Meter 1
Using a local exploit of OS of Meter 1
Availability of Physical Layer of Meter 1
Availability of OS of Meter 1
Confidentiality of OS of Meter 1
Availability of Meter 1
Confidentiality of Meter 1
AMI Head End sends reading request to AMI Meter
Message 2 between AMI Head End and AMI Meter
AMI meter receives request, performs reading, sends reply
Message 1 between AMI Meter and AMI Head End
AMI Head End receives reply from AMI Meter
Interrupt Link: DCU and Head End
Availability of Link: DCU and Head End
Link: DCU and Head End
Link: Meter 1 and Head End
Breaking into DCU Shelter
Physical Destruction of DCU
Using a local exploit of OS of DCU
Using a remote Exploit of OS of DCU
Availability of Physical Layer of DCU
Availability of OS of DCU
Availability of DCU
Interrupt Link: Meter 1 and DCU
Availability of Link: Meter 1 and DCU
Link: Meter 1 and DCU
Path: Meter 1 and DCU
Interrupt Link: Meter 2 and DCU
Availability of Link: Meter 2 and DCU
Successful Attack using Breaking into House 2
Extract shared key
Confidentiality of Link: Meter 2 and DCU
Confidentiality of Link: Meter 1 and Meter 2
Link: Meter 2 and DCU
Path: Meter 1 and DCU via Meter 2
Breaking into House 2
Physical Destruction of Meter 2
Using a remote Exploit of OS of Meter 2
Using a local exploit of OS of Meter 2
Availability of Physical Layer of Meter 2
Availability of OS of Meter 2
Confidentiality of OS of Meter 2
Availability of Meter 2
Confidentiality of Meter 2
Link: Meter 1 and Meter 2
Interrupt Link: Meter 1 and Meter 2
Availability of Link: Meter 1 and Meter 2

Figure 10: Structure of a GSA-graph generated by our prototype (node labels are not intended to be readable here).

The attacker model input is currently hardcoded in the tool, but we plan to capture it in an XML format as well. Attackers are defined by a combination of access to locations, attacker properties, and attacker action templates.

The evidence input is automatically parsed from CSV files. For each leaf property node, it is possible to provide probability distributions, a probability value, or a reference to an external script. Based on that input, the tool automatically generates the GSA-graph and produces the assessment result. For 10 selected AMI meter-reading test cases from [33], the GSA-graphs have on average 45 nodes, with the largest graph having 130 nodes. The overall assessment process per flow takes around 10 seconds. We also used our tool to verify the example given in Section 4. For this, we specified the example workflow and topology in the same XML formats as our other test cases, and used the tool to automatically generate the GSA-graph. In Fig. 10, we give an high-level overview of the resulting graph.

6. RELATED WORK

In this section, we review related work on security assessment and compare the related work with our proposed workflow-oriented assessment framework.

Traditional methods that do not consider workflows. *Attack trees* [29] and *attack graphs* [28] are two prominent categories of work in the security assessment literature that have been widely used in practice "because they have proved to be an intuitive aid in threat analysis" [21]. In an attack tree, the root node is the global goal of the attacker being modeled, while the children of a node are refinements of this goal. In this way, the attack tree provides an intuitive way to describe the different actions that an attacker might take to achieve his goal. Recently, Kordy et al. extended attack trees to also model the behavior of defenders [15, 16]. Compared with attack trees, attack graphs enable state-based analysis for the protection of assets in networked computer systems (e.g., a file's access rights on a given host). Specifically, attack graphs model how an attacker can compose individual vulnerabilities in a networked system to create a multi-stage attack. LeMay et al. recently extended attack graphs to incorporate the attack behavior of different types of adversaries [18].

In addition to attack trees and attack graphs, a number of other security assessment methods were proposed, both graphical and non-graphical. For example, a Methodically Organized Argument Tree (MOAT) was proposed in [13] to aid software development with explicit considerations for system security. The MOAT consists of a collection of assurance arguments represented by trees with logical AND/OR operations for aggregation. Security assurance cases [3, 6, 11] were proposed to demonstrate the security attributes of a given system by putting together a structured collection of security-related claims, arguments, and evidence. In particular, the graphical Goal Structuring Notation (GSN) is often adopted to present security assurance cases [11]. Attack surface analysis [19, 20] aims to measure unknown vulnerabilities for a given software system, by identifying the resources (including methods, channels, and data items) that an attacker can potentially exploit.

The above traditional security assessment methods focus on evaluating the protection of assets or the security properties of target systems. They do not consider workflow information that models how different components interact with each other to provide intended functionality—a feature we feel is important for the assessment of complex systems.

Recent studies that explicitly consider workflows. In the context of voting processes, some recent works have started to explicitly consider the use of workflow information [17, 27, 32]. For example, in [17], the authors first develop a reusable threat model of election based on attack trees, then they customize the tree according to the specific voting workflow information in Marin County. Using an associated software tool, the resulting attack trees can then be evaluated to find attacks with minimal cost in terms of persons involved (attack team size). In addition to assessing the security of voting, workflow information has also been used in assessing the security of banking [12] and cyber-physical systems [2]. While these recent efforts explicitly consider workflow in security assessment, their approaches are to incorporate workflow information into an existing security assessment method (e.g., attack tree in [17], and risk analysis in [2, 12]), so that they can either consider workflow information when applying the existing security assessment method, or they can aggregate the results from existing assessment methods to compute some workflow-level security metrics.

In this work, we take this direction one step further and propose the use of workflows as the central backbone of the security assessment. This structure will allow a tool to automatically identify relevant systems and attacker information as security evidence, and aggregate the related evidence in an organized way to construct a holistic workflow-centric argument graph. In other words, our work is a new type of security assessment method centered around workflow, instead of an amendment to existing security assessment methods to incorporate workflow information. As such, our framework can potentially achieve better generality and reusability by leveraging the generality of workflow specifications themselves. Specifically, a workflow can model various interactions between different types of entities, irrespective of whether the interactions are messages sent over computer networks, communications between humans, or actions with physical consequences, e.g., a governor changing the valve of a steam engine.

Compared to these recent research efforts that extend existing assessment methods with workflow information, our approach of redesigning the assessment process around workflow leads to the following potential benefits: (i) Our formalism, i.e., the GSA argument graph, enables us to organize different types of security-related information into a clear argument structure that centers around workflow. (ii) If machine-readable workflow specifications of the target system are available, our tool can automatically use them to generate the backbone of our graph and suggest relevant supporting information. This can greatly reduce the manual effort for the assessment. (iii) When constructing more detailed argument graphs, our tool allows different types of templates to be applied. For a specific context (e.g., election), domain-specific templates can be developed and reused across different specific scenarios. Furthermore, we envision that many common templates (e.g., regarding message passing, system redundancy, or component trustworthiness) can be reused across very different domains.

7. CONCLUSION AND FUTURE WORK

In this paper, we consider cybersecurity assessment of complex computing systems, including cyber-physical systems. Specifically, we propose a novel approach that takes into account a detailed system description, attacker-specific information, and workflow description to reason about security properties in a holistic manner while facilitating automation. By "going with the flow" and using the workflow as backbone of the argument graph, our assessment framework can integrate the diverse inputs to represent complex systems. The proposed reasoning framework is applied to evaluate the availability and confidentiality of an example Advance Metering Infrastructure. The assessment process is currently being implemented in a tool.

In our future work, we plan to extend the prototype tool to fully implement all aspects of the framework so that it can be applied to a broad range of systems and application domains. As one of those extensions, we will adapt the tool to handle more complex input descriptions. For example, in our case study (see Section 4), we consider a single workflow. By supporting multiple parallel workflows, our framework will be able to accurately assess more complex system configurations, application scenarios, and attacker models. For example, our framework could assess the susceptibility of a power grid system to a blackout attack. In such a scenario, the framework would allow one to consider (as part of the security assessment) the random delay of the meter deactivation and customer feedback, which is needed to accurately assess the threat of the described attack.

8. ACKNOWLEDGMENTS

This work is supported by Singapore's Agency for Science, Technology, and Research (A*STAR) under the Human Sixth Sense Programme (HSSP). We thank our shepherd Sam Weber for his helpful feedback, and Jenny Applequist for her help with editing.

9. REFERENCES

[1] R. Anderson and S. Fuloria. Who controls the off switch? In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, 2010.

[2] Z. Anwar, R. Shankesi, and R. H. Campbell. Automatic security assessment of critical cyber-infrastructures. In *Proc. of the Conference on Dependable Systems and Networks (DSN)*, 2008.

[3] R. Bloomfield, S. Guerra, M. Masera, A. Miller, and O. Saydjari. Assurance cases for security. In *Proc. of the Assurance Case Workshop*, 2005.

[4] V. Cortellessa and V. Grassi. Reliability modeling and analysis of service-oriented architectures. In L. Baresi and E. D. Nitto, editors, *Test and Analysis of Web Services*, pages 339–362. Springer, 2007.

[5] D. Eskins. *Modeling Human Decision Points in Complex Systems*. PhD thesis, UIUC, 2012.

[6] J. Goodenough, H. Lipson, and C. Weinstock. Arguing security—creating security assurance cases. available online: `https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/assurance/643-BSI.html`, last updated June 21, 2012.

[7] D. Grochocki, J. H. Huh, R. Bobba, W. H. Sanders, A. A. Cardenas, and J. G. Jetcheva. AMI threats, intrusion detection requirements and deployment recommendations. In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, 2012.

[8] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wireless Networks*, 11(1-2):21–38, Jan. 2005.

[9] ICS-CERT. CSET: The cyber security evaluation tool. `ics-cert.us-cert.gov/satool.html`, last accessed on April 11, 2013.

[10] K. Ingols, R. Lippmann, and K. Piwowarski. Practical attack graph generation for network defense. In *Proc. of the Annual Computer Security Applications Conference (ACSAC)*, 2006.

[11] T. Kelly and R. Weaver. The goal structuring notation—a safety argument notation. In *Proc. of Dependable Systems and Networks Workshop on Assurance Cases*, 2004.

[12] K. Khanmohammadi and S. Houmb. Business process-based information security risk assessment. In *Proc. of the Conference on Network and System Security (NSS)*, 2010.

[13] D. M. Kienzle and W. A. Wulf. A practical approach to security assessment. In *Proc. of the New Security Paradigms Workshop (NSPW)*, 1997.

[14] C. Ko, M. Ruschitzka, and K. Levitt. Execution monitoring of security-critical programs in distributed systems: a specification-based approach. In *Proc. of the IEEE Symposium on Security and Privacy*, May 1997.

[15] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer. Foundations of attack-defense trees. In *Proc. of the conference on Formal Aspects of Security and Trust (FAST)*, 2011.

[16] B. Kordy, S. Mauw, and P. Schweitzer. Quantitative questions on attack-defense trees. In *Proc. of Conference on Information Security and Cryptology (ICISC)*, 2012.

[17] E. L. Lazarus, D. L. Dill, J. Epstein, and J. L. Hall. Applying a reusable election threat model at the county level. In *Proc. of the conference on Electronic voting technology/workshop on trustworthy elections (EVT/WOTE)*. USENIX Association, 2011.

[18] E. LeMay, M. Ford, K. Keefe, W. Sanders, and C. Muehrke. Model-based security metrics using ADversary VIew Security Evaluation (ADVISE). In

Proc. of the Conference on Quantitative Evaluation of SysTems (QEST), 2011.

[19] P. Manadhata, K. Tan, R. Maxion, and J. Wing. An approach to measuring a systems attack surface. In *Technical report, School of Computer Science, Carnegie Mellon University*, 2007.

[20] P. K. Manadhata and J. M. Wing. An attack surface metric. *IEEE Trans. Software Eng.*, 37(3):371–386, 2011.

[21] S. Mauw and M. Oostdijk. Foundations of attack trees. In *Proc. of Conference on Information Security and Cryptology (ICISC)*, 2005.

[22] S. McLaughlin, D. Podkuiko, and P. McDaniel. Energy theft in the advanced metering infrastructure. In *Proc. of Critical Information Infrastructures Security (CRITIS)*, 2010.

[23] N. Milanovic and B. Milic. Automatic generation of service availability models. *IEEE Trans. on Serv. Comp.*, 4(1):56–69, 3 2011.

[24] K. Muniswamy-Reddy, U. Braun, D. Holland, P. Macko, D. Maclean, D. Margo, M. Seltzer, and R. Smogor. Layering in provenance systems. In *Proc. of the USENIX Annual Technical Conference*, June 2009.

[25] D. Nicol, W. Sanders, and K. Trivedi. Model-based evaluation: from dependability to security. *IEEE Transactions on Dependable and Secure Computing*, 1(1):48–65, 2004.

[26] X. Ou, W. Boyer, and M. McQueen. A scalable approach to attack graph generation. In *Proc. of the Conference on Computer and Communications Security (CCS)*. ACM, 2006.

[27] H. Phan, G. Avrunin, M. Bishop, L. A. Clarke, and L. J. Osterweil. A systematic process-model-based approach for synthesizing attacks and evaluating them. In *Proc. of the international conference on Electronic Voting Technology/Workshop on Trustworthy Elections (EVT/WOTE)*. USENIX Association, 2012.

[28] C. Phillips and L. Swiler. A graph-based system for network-vulnerability analysis. In *Proc. of the New Security Paradigms Workshop (NSPW)*, 1998.

[29] B. Schneier. Attack trees: Modeling security threats. *Dr. Dobb's Journal*, 1999.

[30] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing. Automated generation and analysis of attack graphs. In *Proc. of the IEEE Symposium on Security and Privacy*, 2002.

[31] F. Siddiqui, S. Zeadally, C. Alcaraz, and S. Galvao. Smart grid privacy: Issues and solutions. In *Proc. of Conference on Computer Communications and Networks (ICCCN)*, 2012.

[32] B. I. Simidchieva, S. J. Engle, M. Clifford, A. C. Jones, S. Peisert, M. Bishop, L. A. Clarke, and L. J. Osterweil. Modeling and analyzing faults to improve election process robustness. In *Proc. of the international conference on Electronic voting technology/workshop on trustworthy elections (EVT/WOTE)*. USENIX Association, 2010.

[33] Smartgridipedia. online, `http://www.smartgridipedia.org`.

[34] W. G. Temple, B. Chen, and N. O. Tippenhauer. Delay makes a difference: Smart grid resilience under remote meter disconnect attack. In *Proc. of the IEEE Conference on Smart Grid Communications (SmartGridComm)*, 2013.

[35] U.K. Gov. Dept. of Energy & Climate Change. Government response to the consultation on the second version of the smart metering equipment technical specifications, 2013.

[36] U.S. Department of Energy. 2010 smart grid system report. `http://energy.gov/oe/downloads/2010-smart-grid-system-report-february-2012`.

[37] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl. *Fault Tree Handbook*. U.S. Nuclear Reg. Comm., 1981.

[38] L. Wang, A. Singhal, and S. Jajodia. Measuring the overall security of network configurations using attack graphs. In *Proc. of IFIP WG 11.3 Working Conference on Data and Applications Security*. Springer, 2007.