Formal Synthesis of Dependable Configurations for Advanced Metering Infrastructures

Mohammad Ashiqur Rahman* and Ehab Al- Shaer†
*Department of Computer Science, Tennessee Tech University, USA
†Department of Software and Information Systems, University of North Carolina at Charlotte, USA
Emails: marahman@tntech.edu, ealshaer@uncc.com

***Abstract*—The Advanced Metering Infrastructure (AMI) in a smart grid comprises of a large number of smart meters along with heterogeneous cyber-physical components. These components communicate with each other through different communication media, protocols, and delivery modes for transmitting usage reports and control commands to and from the utility. There is potential for dependability threats especially due to misconfigurations, which can easily disrupt the operations in AMI. Therefore, an AMI must be configured correctly. In this paper, we present an automated configuration synthesis framework that mitigates potential threats by eliminating misconfigurations. We have manifold contributions in this research: (i) formal modeling of AMI configurations including AMI device configurations, topology and communication properties, and data flows among the devices; (ii) formal modeling of AMI operational integrity properties considering the interdependencies among AMI devices' configurations; and (iii) implementing the model using Satisfiability Modulo Theories (SMT), execution of which synthesizes necessary AMI configurations. We demonstrate the proposed framework on an example case study and evaluate the scalability of the framework on various synthetic AMI networks.**

***Keywords*—Advanced metering infrastructure; configuration synthesis; dependability; formal model.**

I. Introduction

AMI is a core component of a smart grid and it provides two-way communication between smart meters and the utility system (particularly, one or more headend servers) through intelligent collectors, which allows energy service providers to monitor and control power consumption remotely. These devices communicate with one another using different communication media, protocols, and security policies. Energy usage data is transferred from meters to collectors and from collectors to the headend server following different data delivery modes under the control of various security and business policies.

Misconfigurations can cause nontrivial threats to the security and reliability of an AMI system because of interdependencies among device configurations, communication and security properties, and mission requirements. On the other hand, due to the limited budget and benefit, it is not possible to deploy so many devices (*e.g.*, collectors) in order to have a dependable data delivery in AMI. Manual enforcement of the appropriate AMI configurations can be overwhelming and often inaccurate due to high potentiality of human errors. Therefore, there is a pressing need for the automatic synthesis of the AMI topology and devices' configurations, ensuring operational integrity of the system. In this paper, we address this need by presenting an automated configuration synthesis framework for an AMI system. In this framework, we create a logic-based formal model of the AMI topology and devices' configurations, data delivery among the devices, and operational integrity requirements. The solution to this formal model synthesizes AMI configurations, including a deployment or placement design for collectors and report schedules for meters and collectors. We apply abstraction in modeling smart meters and their association with collectors, which allows the proposed synthesis mechanism to scale with large numbers of smart meters. We implement the framework and illustrate its execution using an example case study. We use SMT (Satisfiability Modulo Theories) to formalize the framework [1]. SMT consists of powerful logic theories that can solve hard constraint satisfaction problems which arise in many diverse areas, including software and hardware verification, test-case generation, scheduling, and planning. We also evaluate the accuracy and scalability of our framework by running it on various synthetic test networks.

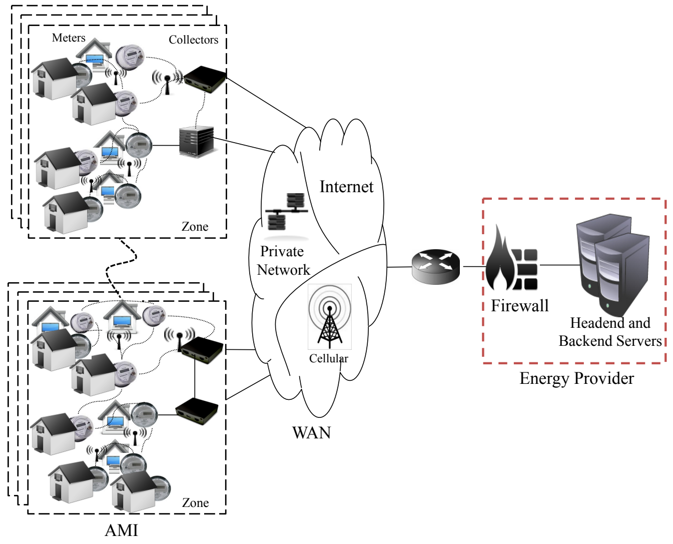

Fig. 1. The topology of a typical advanced metering infrastructure.

The rest of the paper is organized as follows. We discuss the motivation of our research in Section II. We present the architecture of synthesis framework and the corresponding formal model in Section III. The evaluation results are presented in Section IV. The related work is briefly discussed in Section V, which is followed by the conclusion.

II. Background and Challenges

A. Advance Metering Infrastructure

The typical network structure of an AMI system is shown in Fig. 1 [2], [3]. An AMI system often consists of thousands of smart meters and hundreds of intelligent data concentrators or collectors. A meter reports energy usage data to a specific collector periodically. A collector stores the data received from a group of meters in its buffer and forwards the stored data to a server located at the energy provider's utility network. This server is often known as the headend system. Although in some AMI architectures, a meter directly reports energy usage data to the headend system, often data collectors are used to collect and store meter data and later to send the stored data to the headend system when it is required [4], [2], [3], [5]. This collector based AMI design gives better manageability by allowing scalable infrastructure design, flexible protocol use, and efficient networking. The collector also forwards control commands and patches from the headend to the meters. A meter is connected to a collector either directly or through another meter. The latter case occurs in a mesh network of meters, where intermediate meters relay the data to the collector. Collectors are connected to a headend usually through a proprietary but often a third party network. Unlike the policy-based Internet forwarding, data deliveries in an AMI network are either time-driven or request-driven and they follow specific schedules. In the time-driven or push-driven mode a meter or a collector reports data periodically based on a pre-configured delivery schedule, while in the request-driven or pull-driven mode a meter or a collector reports data only upon receiving a request. In the pull-driven mode, requests are often sent periodically following a schedule [2], [6]. In practice, the push mode is used between meter and collector, while the pull mode is used between collector and headend. For the purpose of successful delivery of data, an AMI network must be configured carefully to synchronize the data delivery without overflowing the network or its devices.

B. Causes of Threats to AMI

There are two main causes of threats to AMI [7], [8]: (i) the presence of vulnerabilities in the system and (ii) the lack of preparedness against attacks. Misconfiguration is one of the major sources of vulnerabilities in AMI. For example, improper data scheduling due to misconfiguration can cause data loss by overflowing the communication bandwidth or the collector's storage capacity. Similarly, due to communication protocol or security policy misconfiguration, reachability or trusted communication can be failed. It is well documented that configuration errors cause 50%-80% of vulnerabilities in cyber infrastructure [9]. Let us provide an example of a misconfiguration scenario. Although the values used in this example are synthetic, they are motivated from [2], [5], [10]. A collector receives reports from 100 meters of two types. Each meter of one kind has a sampling rate of 18 KB per 30 seconds, while each meter of another kind has a sampling rate of 20 KB per 40 seconds. Among these 100 meters, 60 of them fall into the first kind, while the rest of them fall into the second kind. Therefore, the collector will receive 3,360 KB (in an average) data in every 60 seconds, which is to be stored in its buffer. The size of the collector's buffer is 80 MB ($\approx$ 80,000 KB). Let us assume that the collector is configured with the report schedule, according to which the collector sends the data to the headend system in every 1440 seconds. Thus, in this reporting interval, 80,640 KB of total data will be reported to the collector by these meters. It is obvious that this amount of data (80,640 KB) will flood the collector's buffer (80,000 KB), which will in turn cause data loss (*i.e.*, initial 640 KB report data will be overwritten).

It is worth mentioning the present practice of billing data collection. The data collection from the meters/collectors is very often once or twice a day, in better cases hourly, while in worse cases monthly [11]. Moreover, there is no or very few demand-response management services running in smart grids at present as the projects of deploying smart meters are still continuing [10], [12]. In near future when the ideal objectives of the AMI system will start to be performed in practice, the data collection rate will increase significantly, even per second. We believe that, at that time, the issue of data loss during data collection due to lack of resources (data collectors), attacked/failure incidents, or improper reporting or data collection scheduling will be very critical leading to disrupted services, decreased revenue, and loss of reputation.

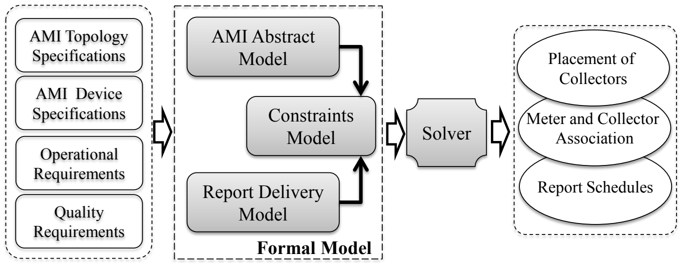

Fig. 2. The AMI Synthesizer Framework.

C. Objective

The correct functioning of an AMI system stands on consistent and secure execution of tasks in time. The reliability of configuration depends not only on the local device parameters but also on the safe and secure interactions of these devices across the network. There is a significant number of logical constraints on configuration parameters of thousands of AMI devices, which need to be satisfied to ensure dependable communications among AMI components. These constraints represent system invariants and user-driven (*i.e.*, organizational) requirements. There is no such formal framework for automatically configuring an AMI system based on the essential and organizational dependability requirements. In this work, our objective is to develop an automated framework that takes necessary inputs, such as AMI topology and device specifications, operational integrity properties, user-driven requirements (*e.g.*, data freshness), resource constraints (*e.g.*, deployment budget), and automatically synthesizes necessary AMI configurations satisfying the given properties and requirements.

III. Formal Model of the Synthesis Framework

A. Synthesis Framework Architecture

The architecture of the AMI configuration synthesis framework is shown in Fig. 2. The framework follows a top-down design automation approach instead of the traditional bottom-up verification-based design approach. In this framework, we first formally model the AMI system, including the topology, devices, and data deliveries. Then, we formally model operational integrity requirements on top of the AMI system model, satisfaction of which determines necessary AMI configurations, which include an appropriate deployment of AMI devices (particularly collectors) and their report schedules. We Implement the model using SMT.

The synthesis framework takes different inputs as shown in Fig. 2: (i-ii) specifications about the AMI topology and devices (smart meters and collectors), (iii) AMI operational invariants, and (iv) user-driven requirements and constraints that mainly include data freshness requirements and the deployment budget constraint. In this work, for a particular AMI topology, smart meters are considered as already deployed, while the collectors are required to be deployed satisfying dependability properties. With respect to the inputs, the framework models the deployment of collectors, reachability among the devices, report schedules, topological and operational invariants, and other user-driven requirements, and resource limitations as constraints. The solution to this model satisfies the constraints and provides AMI topology configurations (*i.e.*, deployment design for necessary collectors) and AMI devices' configurations (*i.e.*, report schedules of the meters and collectors). The deployment plan for collectors includes where (placements) and what (collector types and their numbers) to deploy.

B. AMI Configurations Parameters

We define a number of parameters to denote AMI configurations that includes AMI devices and topology properties. In our notations, variables start with small alphabetic letters, while constants start with capital letters. In this paper, we use multiple-letter notations to denote many parameters. We expect that these multiple-letter notations will help the readers to recall them. Also note that, no multiplication of two parameters is represented here without the multiplication sign.

Configuration Level Abstraction

An enterprise AMI network typically consists of thousands of smart meters distributed over different geographical regions. The meters communicate with collectors for delivering data based on device configurations and communication properties. For the purpose of achieving better scalability, we apply the concept of abstraction in terms of groups or classes based on the similarities between the configurations of the meters. A particular group or class of devices shares the same (physical and logical) configuration properties. Collectors are modeled as individual devices. Moreover, we use the term *zone* to denote a collection of meters residing at the same geographical area. The meters within a zone form a mesh network to communicate to collectors deployed in that zone. This collection of meters often forms a number of meter groups. Therefore, a meter group is identified or localized with respect to a zone.

AMI Device Configurations:

Smart Meter: A meter group is identified by $m_{k,i}$, where k is the zone index and i is the meter group index. A meter group exists when $m_{k,i}$ is true. The objective of our synthesis framework is to synthesize properties of each existing meter group. We use $mT_{k,i}$ for denoting the meter type and $mS_{k,i}$ for representing the group size. A particular type of meter is mainly specific to a vendor and it has a specific data sampling rate (*i.e.*, the number of samples per time slot), as well as a specific size for each sample. Since each meter in a group has the same type, they have the same property values. That is, the sampling rate and the sample size of each meter of a group are the same and they are denoted by $mSR_{k,i}$ and $mSS_{k,i}$, respectively. The report schedule is represented by two parameters, the base (starting) time of reporting ($mRB_{k,i}$) and the reporting interval ($mRI_{k,i}$), which indicate that the meters of this group report periodically at each interval starting from the base time with respect to a specific time period, *e.g.*, during a day. We use $mC_{k,j}$ to identify the collector that is associated to the meter group. We assume minute as the unit for time slots and kilo bytes (KB) for the data or storage size.

Intelligent Data Collector: A collector is represented by $c_{k,j}$, where k is the zone index and j is the collector index. If $c_{k,j}$ is true, then the associated collector is deployed in the system. The objective of our synthesis framework includes synthesizing the collectors to be deployed in each zone, their configurations, and the association between the collectors and the meter groups in each zone. We use $cT_{k,j}$ to denote the collector type. A particular type of collector has a specific buffer size and it is associated with a deployement cost (price). The buffer size is denoted by $cBS_{k,j}$ and the deployment cost by $cC_{k,j}$. Similar to a meter, the report schedule of a collector is also represented by two parameters: the base time $cRB_{k,j}$ and the reporting interval $cRI_{k,j}$.

AMI Topology Configurations:

An AMI topology mainly defines the connectivity between the AMI devices. As shown in Fig. 1, the AMI topology, *i.e.*, the connectivity between the AMI devices, is well defined. The meters in a particular zone are considered to be connected to one or more collectors by forming a mesh network between them. We consider MBW as the average bandwidth of the mesh network communication. A collector can be individually connected to the headend system through WAN communication. If a collector is not connected with the headend system directly, it needs to be connected with another collector (cFC) to forward its stored report data to the headend. All of the collectors connected to the headend share the path in the energy provider's network after the border router. The individual paths from the collector to the border router can be wired, wireless, or cellular. We use cIP to denote the (individual) communication path type deployed for a collector toward the headend. Each path type has a specific bandwidth (PB) and a deployment cost (PC). We use SBW to denote the shared link bandwidth from the border router to the headend. The bandwidths of these communication paths play an important role for choosing the report schedules.

C. Modeling of AMI Configurations

The constraints associated with AMI dependability requirements are divided into system invariants (*i.e.*, operational integrity constraints) and user-driven requirements (data freshness constraints). Modeling of system invariants includes modeling of configurations for meters and collectors and that of report schedules.

Meter Groups and their Properties:

The meters in a group have the same meter type. A valid meter type (between 1 to MT number of available types) for group i in zone k is ensured as follows:

$$m_{k,i} \rightarrow (mT_{k,i} \geq 1) \land (mT_{k,i} \leq MT)$$

We assume that the meters are already deployed. That is, the number of a particular type of meters in a zone is given. Since a meter group in a zone has a specific type, the size of the group must be within the number of meters of that particular type residing in the zone:

$$m_{k,i} \rightarrow (mT_{k,i} = t) \rightarrow mS_{k,i} \geq 1 \land mS_{k,i} \leq MS_{k,t}$$

In the above constraint formulation, $MS_{k,t}$ denotes the number of meters of type t residing in zone k. Moreover, if we sum up the sizes of all meter groups in a zone having the same meter type, then the summation must be equal to the total of this particular type of meters in the zone. The sampling rate and the sample size of each meter of a meter group in a zone depend on its type. If MSR_t and MSS_t are the sampling rate and sample size of a meter of type t, then:

$$(mT_{k,i} = t) \rightarrow (mSR_{k,i} = MSR_t) \wedge (mSS_{k,i} = MSS_t)$$

The meters of a meter group in a zone send their sampled data to a specific collector deployed in the same zone. If CN is the maximum number of potential collectors in a zone, then:

$$m_{k,i} \rightarrow (mC_{k,i} \geq 1) \wedge (mC_{k,i} \leq CN)$$

In a particular zone, no two meter groups can have the same values for all properties. That is:

$$\begin{aligned} &m_{k,i} \wedge m_{k,\hat{i}} \wedge (i \neq \hat{i}) \rightarrow \\ &\quad \neg((mT_{k,i} = mT_{k,\hat{i}}) \wedge (mC_{k,i} = mC_{k,\hat{i}}) \\ &\quad \wedge (mRB_{k,i} = mRB_{k,\hat{i}}) \wedge (mRI_{k,i} = mRI_{k,\hat{i}})) \end{aligned}$$

Collectors and their Properties:

There is a finite number of collector types. Let this number be CT. A collector's type must be within this set of types. Therefore, if collector j exists in zone k, then its type should satisfy the following constraint:

$$c_{k,j} \rightarrow (cT_{k,j} \geq 1) \wedge (cT_{k,j} \leq CT)$$

The buffer size and deployment cost of each collector in a zone depend on its type. Therefore, if CBS_t and CC_t are the buffer size and the deployment cost of a collector of type t, respectively, then we have the following:

$$(cT_{k,j} = t) \rightarrow (cBS_{k,j} = CBS_t) \wedge (cC_{k,j} = CC_t)$$

If a collector is selected as the designated collector for a meter group in a zone for reporting, that indicates that the particular collector is deployed. Therefore:

$$\bigvee_i (mC_{k,i} = j) \rightarrow c_{k,j}$$

Topology and its Properties:

If PT is the number of communication path types, the type of collector j's individual communication path must be within this set of types, while the type is zero when there is no path:

$$c_{k,j} \rightarrow (cIP_{k,j} \geq 0) \wedge (cIP_{k,j} \leq PT)$$

The bandwidth ($cPB_{k,j}$) of the path and its deployment cost ($cPC_{k,j}$) correspond to those of the path type. If collector j does not have communication path to the headend system, it must be connected to a collector (in the same zone) to forward its report data to the headend:

$$(cIP_{k,j} = 0) \rightarrow (cFC_{k,j} \geq 1) \wedge (cFC_{k,j} \leq CN)$$

The forwarding collector should be a deployed one and it must have a communication path to the headend:

$$(cFC_{k,j} = \hat{j}) \rightarrow \exists_{\hat{j}}\ c_{k,\hat{j}} \wedge (cIP_{k,\hat{j}} > 0)$$

Report Schedule Constraints:

We consider finite sets of potential values for the base time of the report schedule for meters ($\mathcal{B}_M$) and collectors ($\mathcal{B}_C$). Therefore, $mRB_{k,i} \in \mathcal{B}_M$ and $cRB_{k,j} \in \mathcal{B}_C$. Similarly, we consider a finite set of potential values for reporting intervals. We have a number of invariant constraints to choose the reporting schedules. First, the base time of a report schedule must be lower than its interval. Second, a collector should report less frequently than its associated meters, so that no reporting is done without new data. The following equations ensure these constraints:

$$\begin{aligned} &m_{k,i} \rightarrow mRB_{k,i} < mRI_{k,i} \\ &c_{k,j} \rightarrow cRB_{k,j} < cRI_{k,j} \\ &(mC_{k,i} = j) \rightarrow mRI_{k,i} \leq cRI_{k,j} \end{aligned}$$

A collector should forward its stored usage data to the headend system in a timely manner so that no part of this data is overwritten by new data. That is, the total incoming data from the meters within the report interval of the collector should not exceed its buffer. Moreover, the reporting data should not exceed the communication bandwidth. Let mRA denote whether a meter reports at a particular time slot (s) and mRS denote the size of the reported data. We have similar parameters (cRA and cRS) for collectors. We calculate mRA (for each zone k, meter group i, and slot s) as follows:

$$mRA_{k,i,s} \rightarrow m_{k,i} \rightarrow ((s - mRB_{k,i}) \% mRI_{k,i} = 0)$$

Similarly, we calculate cRA for collectors. We compute the report size of a meter considering the average number of times a meter sends data to the associated collector within the reporting interval of the meter:

$$m_{k,i} \rightarrow (mRS_{k,i} = mS_{k,i} \times mSS_{k,i} \times mRI_{k,i} / mSR_{k,i})$$

The report data for a collector (cRS) is ultimately the total usage data sent to the collector by the associated meters:

$$c_{k,j} \rightarrow (cRS_{k,j} = \sum_{\{i | (mC_{k,i} = j)\}} mRS_{k,i} \times cRI_{k,j} / mRI_{k,i})$$

The following equation ensures no overwrite on the stored data in the collector's buffer:

$$c_{k,j} \rightarrow cRS_{k,j} \leq cBS_{k,j}$$

We assume that the bandwidth in a mesh network is shared by the participating nodes in the network. Therefore, to ensure the successful delivery of usage data to a collector from the associated meters, the accumulated rate of data transmission by the meters must be within the bandwidth:

$$\sum_{\{i | m_{k,i}\}} mRS_{k,i} \leq MBW \times mRI_{k,i}$$

If collector j in zone k is not connected to the border router of the energy provider's network directly, it sends the data to a neighboring collector ($cFC_{k,j}$) according to its own schedule. We assume that the communication latency between these two collectors are negligible compared to the long distance to the headend. Thus, for each time slot (s), the communication bandwidth constraint for the communication from collector j to the border router is formulated considering the reporting schedules of itself and the collectors forwarding to it, and the associated report sizes:

$$\begin{aligned} (cRA_{k,j,s} \times cRS_{k,j}) + \sum_{\hat{j}, cFC_{k,\hat{j}} = j} (cRA_{k,\hat{j},s} \times cRS_{k,\hat{j}}) \\ \leq cPB_{k,j} \times cRI_{k,j} \end{aligned}$$

Since the path after the border router to the headend is shared by all of the collectors, we formalize a similar bandwidth constraint by summing up all the reports at a particular

TABLE I. Example's Output (Meters' Configurations)

Zone	Group Id	Meter Type	Group Size $\times 20$	Associated Collector Id	Reporting Base Time	Reporting Interval
1	1	1	9	1	60	120
1	2	2	8	3	0	30
1	3	1	21	2	30	60
…	…	…	…	…	…	…
2	7	2	10	1	0	30
2	9	1	12	2	10	120
3	8	1	20	2	0	120
3	9	2	26	3	60	120
…	…	…	…	…	…	…

time slot. Moreover, we also consider different organizational constraints. For example, the data transmission delay from a meter to a collector should reach within a threshold time.

D. An Example Case Study

We implement our synthesis model using Z3 [1]. Here, we illustrate the execution of the model with a synthetic and small example. We consider an arbitrary AMI system of 2,000 smart meters distributed in 4 zones. It is required to find safe and reliable configurations of the AMI system, including the deployment of collectors, association among meters and collectors, and report schedules within the given deployment budget. There are 2 types of meters and the number of each type in a specific zone is given. Each type of meter has a particular set of properties (*i.e.*, sampling rate and size of each sample). A type 1 meter takes a sample (of size 2 KB) at each 5 minutes, while a type 2 meter takes a sample (of size 3 KB) at each 10 minutes. The minimum reporting interval for a meter is considered as 30 minutes, while the maximum is 120 minutes, while they are 120 minutes and 360 minutes for collectors. The maximum number of meter groups expected in a zone is 10, while each group should have at least 20 meters. A collector can be either of 2 types, while each type has a different buffer size (10,000 KB and 12,000 KB buffer, respectively) and deployment cost ($7K and $10K, respectively). The maximum number of collectors that can be deployed in a zone is 5. The bandwidth of the mesh network between meters and collectors is 200 kbps. There are three options for the individual path from a collector to the utility's border router (bandwidths of 100, 200, and 1000 kbps and deployment costs of $10K, $15K, and $20K), while the shared link after the utility border router toward the headend system is 10000 kbps. According to the data freshness constraint, the data should reach from a meter to a collector in 5 minutes, while from a collector to the headend system in 30 minutes. The deployment budget is $200K.

The formal model corresponding to this example returns a satisfiable result including necessary configuration parameters. The configurations associated with the meters and the deployed collectors are shown (partially) in Tables I and II. We see that 8 meters groups are selected in zone 1, 3 groups in zone 2, 2 groups in zone 3, and 5 groups in zone 4. The collector's id associated to each meter group is identified (*e.g.*, in zone 2, meter group 7 is associated with collector 1). With regards to the collector deployment, 4 collectors are selected to be deployed in zone 1, 2 collectors are selected for each of zones 2 and 3, and 3 collectors in zone 4. Collectors 2 and 3 in zone 1, collector 1 in zone 2, collectors 2 and 3 in zone 3, and collector 2 in zone 4 have communication paths to the headend. The report schedules are selected in such a way that the collectors do the reporting in distributed time slots, considering the limited buffer sizes and bandwidths.

TABLE II. Example's Output (Collectors' Configurations)

Zone	Collector Index	Collector Type	Reporting Base Time	Reporting Interval	Comm Path Type	Forwarding Collector
1	1	2	60	120	-	2
1	2	2	60	120	2	-
1	3	1	120	240	2	-
1	4	1	120	240	-	3
2	1	2	60	120	1	-
…	…	…	…	…	…	…

IV. Evaluation

We evaluate our AMI synthesis framework mainly in terms of scalability. We analyze the tool by evaluating different constraints under synthetic AMI configurations.

Methodology: We evaluate the scalability of the AMI configuration synthesis model by analyzing the time and memory required in constraint verification by varying the AMI network size. We consider the network size as the total number of smart meters in the AMI system, which are distributed in different sizes of zones. Since an organization usually is limited within the choice of a few types of meters and collectors, we consider up to 3 types of meters or collectors in our experiments. The number of potential values of the reporting base time as well as the interval is kept less than or equal to 10. We run our experiments on an Intel Core i5 machine with 8 GB memory.

Impact of the Problem Size on Execution Time: Fig. 3(a) shows the execution time of our synthesis framework with respect to the AMI size, *i.e.*, the number of smart meters. We show the execution time in two different scenarios of the number of collector types. The graphs in the figure show that with the number of meters, the increase in the execution time lies between linear and quadratic growths. Although the number of parameters seems to be increased exponentially (as does the execution time), we observe complexity less than that. This is due to the property-based abstraction that is applied in modeling. The evaluation results with respect to the average size of each zone are shown in Fig. 3(b). The graphs shows that the execution time decreases with the zone size. If the size increases, the number of zones reduces in the AMI network, which ultimately reduces the effective problem size.

Impact of the Constraints on Execution Time: The synthesis of AMI configurations depends on the given constraints (*e.g.*, budget and data freshness constraints). The tighter the constraint, the more time is required to synthesize the configurations. We analyze the impact of this budget on the execution time. The analysis results are shown in Fig. 3(c) and they show that the execution time increases rapidly with the decrease of the budget. This is because the lower the budget, the more space is required by the solver to search for a satisfiable set of configurations, and thus the execution time increases. If the budget becomes much lower, there may be no solution. In unsatisfiable cases, the execution time is often high, much larger than that in satisfiable cases, as whole of the search space needs to be traversed to conclude that there is no solution. However, if a constraint is too tight (*e.g.*, the budget is too low), the solver takes a much shorter time to conclude with unsatisfiability because the search space becomes small due to the extremely tight constraints.

Memory Requirement: We evaluate the memory requirement for executing our model in the SMT solver [1] by changing the number of meters. The analysis results are shown in Table III for two different scenarios. In the first scenario, the number of collector types is 2, while in the second scenario, the number

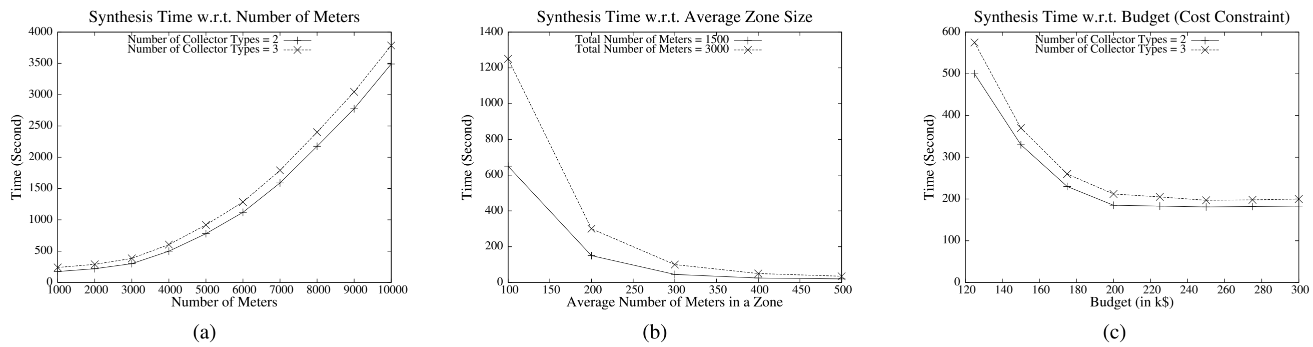

Fig. 3. The execution of the synthesis framework with respect to (a) the number of meters and (b) the average size of each zone, and (c) the impact of the budget constraint on the execution time.

TABLE III. MEMORY REQUIREMENTS (IN MB) W.R.T. PROBLEM SIZE

Hosts	Scenario 1	Scenario 2
1000	43.20	46.20
2000	105.30	110.40
3000	168.10	175.00
4000	330.50	340.90
5000	465.90	478.60

is 3. We observe that the memory requirement lies between the linear and quadratic orders. The table shows that the memory requirement in the second scenario is larger than the memory requirement in the first because, due to a larger number of collector types, there are more options (and so more variables) to design the deployment of collectors.

V. RELATED WORK

Last several years, a significant number of works (*e.g.*, [7], [8]) have been initiated on describing the interoperability among heterogeneous smart grid components including security/dependability issues based on different attack scenarios. McDaniel et al. [13], [14] discussed the security and privacy challenges in smart grid networks. Wang et al. [15] presented an artificial intelligent based approach for analyzing risks in smart grid networks. McLaughlin et al. [16] described an approach for penetration testing on AMI systems. In our previous work [2] we presented a formal model based tool that provably verify operational consistency and security controls in AMI systems. However, all these above mentioned works follow the traditional bottom-up approach of verifying the security/dependability of the system.

The research on the configuration synthesis for dependable cyber and cyber-physical systems is still in the early stage. Narain et al. presented a tool named ConfigAssure in [17], which takes routing specific security requirements and configuration variables as inputs and produces the values of the configuration variables as outputs that make the requirements true. In our previous work [18], we proposed a formal model for generating network security configurations satisfying the given isolation requirements and business constraints. Unlike to all of these works, in this paper, we solve the problem of synthesizing the AMI configurations for its operational integrity, where the dependability requirements are significantly different than that of the traditional networks.

VI. CONCLUSION

In this paper, we present an automated AMI configuration synthesis framework. We model various constraints that are crucial for dependable data delivery in AMI systems. We implement the framework using SMT. The execution of the proposed model synthesizes necessary configurations satisfying the constraints. We evaluate the scalability of our framework in different synthetic AMI networks and requirements and observe that its running time is almost an hour for a network of 10,000 smart meters in our particular computing environment. We achieve significantly high scalability by applying the group-based abstraction in the model.

REFERENCES

[1] Leonardo de Moura and Nikolaj Bjørner. Satisfiability modulo theories: An appetizer. In *Brazilian Symposium on Formal Methods*, 2009.

[2] M. A. Rahman, E. Al-Shaer, and P. Bera. Smartanalyzer: A noninvasive security threat analyzer for ami smart grid. In *31st IEEE INFOCOM*, pages 2255–2263, 2012.

[3] R. Mohassel et al. A survey on advanced metering infrastructure. *Intl. Journal of Electrical Power and Energy Systems*, 63:473 – 484, 2014.

[4] Punya Prakash. Data concentrators: The core of energy and data management. http://www.ti.com/lit/wp/spry248/spry248.pdf?DCMP=induvid1&HQS=ep-pro-sit-induvid1-toolsinsider-20140605-mc-en, 2013. Texas Instruments.

[5] B. Karimi, V. Namboodiri, and M. Jadliwala. On the scalable collection of metering data in smart grids through message concatenation. In *IEEE Intl. Conf. on Smart Grid Communications*, pages 318–323, Oct 2013.

[6] J. Wenger and B.C. Le. System and method to manage utility meter communications, 2014.

[7] AMI-SEC Task Force. Ami system security requirements, 2008. http://osgug.ucaiug.org/utilisec/amisec/.

[8] Smart Grid Interoperability Panel-Cyber Security Working Group. Nistir 7628: Guidelines for smart grid cyber security, 2010.

[9] R. Alimi, Y. Wang, and Y. R. Yang. Shadow configuration as a network management primitive. In *ACM SIGCOMM Conference on Data communication*, pages 111–122, 2008.

[10] HP Technical. Using the hp vertica analytics platform to manage massive volumes of smart meter data. http://www.vertica.com/wp-content/uploads/2014/05/SmartMetering_WP.pdf.

[11] UK Energy Supplier and Energy Company-E.ON. Smart meter frequently asked questions. https://www.eonenergy.com/for-your-home/saving-energy/smart-meters/frequently-asked-questions.

[12] Energy Research Council. Best practices: Demand response. http://energyresearchcouncil.com/best-practices-demand-response.html.

[13] P. McDaniel and S. McLaughlin. Security and privacy challenges in the smart grid. *IEEE Security and Privacy*, 7(3):75–77, 2009.

[14] S. McLaughlin, D. Podkuiko, and P. McDaniel. Energy theft in the advanced metering infrastructure. In *CRITIS*, pages 176–187, 2009.

[15] Y. Wang et al. Computational intelligence algorithms analysis for smart grid cyber security. *Advances in Swarm Intelligence*, 6146:77–84, 2010.

[16] S. McLaughlin et al. Multi-vendor penetration testing in the advanced metering infrastructure. In *ACSAC*, pages 107–116, 2010.

[17] S. Narain, G Levin, S Malik, and V. Kaul. Declarative infrastructure configuration synthesis and debugging. *JNSM*, 16(3):235–258, 2008.

[18] M. A. Rahman and E. Al-Shaer. A formal framework for network security design synthesis. In *33rd ICDCS*, pages 560–570, 2013.