MAR(S)2: Methodology to Articulate the Requirements for Security In SCADA

Tanaya Gopal\textsuperscript{1}, Madhuri Subbaraju\textsuperscript{2}
School of Computer Science
The University of Manchester
Manchester, UK
\textsuperscript{1}tanaya.gopal@postgrad.manchester.ac.uk;
\textsuperscript{2}madhurisubbaraju.nallursubbaraju@postgrad.manchester.ac.uk

Rashmi vivek Joshi\textsuperscript{3}, Somdip Dey\textsuperscript{4}
School of Computer Science
The University of Manchester
Manchester, UK
\textsuperscript{3}rashmivivek.joshi@postgrad.manchester.ac.uk;
\textsuperscript{4}somdip.dey@postgrad.manchester.ac.uk

Abstract— Security requirement is an important aspect of system’s development. There are numerous security requirements methodologies, which have been developed till date. Research is still going on to improve or create new methodologies that will make a system as secure as possible. Asset management, risk assessment, validation of functional and non-functional security requirements and security requirements elicitation are some of the important parts of a security requirements methodology. However most of the security requirements methodologies in use today such as SQUARE, UMLSec, Secure Tropos and CORAS fail to perform one or more of these functions. Additionally, very few methodologies focus on critical infrastructure industrial systems like SCADA. This paper introduces a methodology (MAR(S)2) that incorporates all the important functions, which will produce a strong methodology that produces a profound and well-defined security requirements for SCADA systems.

Keywords—requirements engineering; security; SCADA.

I. INTRODUCTION

Requirements engineering is an important part of software engineering but organisations often fail to gauge its importance. Poor requirements in projects lead to problems such as [1]:

- Budget exceeds the set limit
- Project goes beyond the decided schedule
- System produced is of poor quality
- The scope of the project is reduced
- System is not used after deployment

Many reports have been generated, which cited that poor requirements are one of the top three causes of IT failure [9]. However, security requirements are relatively less among requirements engineering processes and even if they are, they are considered in an ad-hoc manner [9]. Therefore, there is a need to make security requirements an integral part of requirements engineering. This report studies the concept of security requirements engineering and presents an architecture of a security requirements methodology.

A system is only as secure as its weakest part [6] therefore it is very important to look into all the aspects of the security requirements of the system. According to M. Gasser and R Anderson, adding security as an afterthought often leads to problems [6]. Therefore, it should be included at the time of requirements engineering along with the functional and non-functional requirements. We need a requirements methodology that will consider the need for security and make security an important aspect at later stages too. Still a proper security requirements elicitation is not part of the best practice, today [2], which often leads to failure of projects. Therefore, there is a need to formulate precise, complete and testable security requirements at the early stages of software development.

The modern society depends on critical infrastructures such as electricity and water supply, waste management, oil and gas supply systems. They form an important backbone of the society. These systems, when they were developed, were not interconnected and functioned aloof. However, with time these systems have started to become interconnected which has exposed it to many types of cyber and malware attacks [6]. It is therefore, very important to keep them protected against such attacks. These systems are now mostly deployed as SCADA (Supervisory Control and Data Acquisition) systems, which are critical for the functioning of the economy and need high security measures to be implemented for their smooth functioning. Unfortunately, security of these systems was not considered very important until few years ago. It is only recently that people have realised that these systems need strong security measures.

Electric power production and supply is an important example of SCADA systems. The earlier SCADA systems were very basic and consisted of lamps and switches [3]. Nowadays the systems are much more sophisticated and complicated due to the advancement in communication systems. They can be controlled by equipment similar to desktop computers and it does not need to be situated at the same site as the system. That is, it can be controlled by a remote site. This ability has made these systems more prone to security attacks. These attacks can cause significant disruptions if successful [3]. The
August 2003 northeast blackout in US, which also affected Canada [4], shows how important SCADA systems are for an economy. This paper will focus on the security requirements of SCADA systems. There are many different kinds of security requirement methodologies, which are used in organisations or specific industries today. Some of the popular methodologies are SQUARE, UMLSec, Secure Tropos and CORAS. All of these security requirements methodologies include many steps such as establishing terms and definitions, asset management, risk assessment, validation of requirements etc. These are important functions, which ultimately lead to a set of clear, precise and validated security requirements. However, many of them fail to incorporate all of these important functions together. This paper introduces a methodology that will be applied on SCADA systems, which incorporates all the important functions aforementioned.

Section II.A studies different types of security requirements engineering methods. Each of the methodology belongs to a different category and focusses on different areas of security requirements engineering. Section II.B emphasises on the importance of SCADA systems and their security in today’s world. Section III.A compares the four security requirements methodologies mentioned in section II. Section III.B proposes a security requirements methodology that was created as a result of the research and literary review. In section IV the future scope and feasibility of the system is discussed. It ends with a conclusion of the work done.

II. BACKGROUND RESEARCH AND LITERARY REVIEW

For the background research, various security requirements methodologies in use today were studied. Four kinds of methodologies were studied in specific and one was selected from each kind. The selection was made by comparing the amount of research on the methodology done so far or the interest that it has generated. Firstly, it was analysed whether the methodology would work for a security critical system such as SCADA. This was decided by considering whether the methodology could be applied on a large-scale machine or system. The second method was to decide by the amount of interest the methodology has generated in the academic world. This was done by looking at the number of citations of the articles or book that introduced the methodology. Then the final selection of the methodology was made. The methodologies that were considered are shown in the Table 2.1 below:

<table>
<thead>
<tr>
<th>Types</th>
<th>Methodologies</th>
<th>Citations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multilateral Approaches</td>
<td>Multilateral security requirements analysis (MSRA)</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>Security Quality</td>
<td>195</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methodology</th>
<th>Citations</th>
</tr>
</thead>
<tbody>
<tr>
<td>UML Based approaches</td>
<td></td>
</tr>
<tr>
<td>Misuse cases</td>
<td>65</td>
</tr>
<tr>
<td>UMLSec</td>
<td>637</td>
</tr>
<tr>
<td>Secure UML</td>
<td>555</td>
</tr>
<tr>
<td>Goal-oriented approaches</td>
<td></td>
</tr>
<tr>
<td>KAOS+Anti-models</td>
<td>19</td>
</tr>
<tr>
<td>Secure i*</td>
<td>54</td>
</tr>
<tr>
<td>Secure Tropos</td>
<td>132</td>
</tr>
<tr>
<td>Risk analysis based approaches</td>
<td></td>
</tr>
<tr>
<td>CORAS</td>
<td>117</td>
</tr>
<tr>
<td>Tropos Goal risk framework</td>
<td>49</td>
</tr>
<tr>
<td>Model-based information system security risk management (ISSRM)</td>
<td>41</td>
</tr>
</tbody>
</table>

Table 2.1

The four methodologies selected were SQUARE, UMLSec, Secure Tropos and CORAS. The next section explains them in detail.

A. Literary Review

A.1 SQUARE

SQUARE is a security requirements methodology that consists of nine steps to elicit and finalise security requirements for a system. The methodology aims to produce clear, precise, non-conflicting and validated security requirements for a system. Each of the steps requires an input, the techniques for performing that step, participants for the step and the output of the step. Generally, the set of output for one step acts as input for the next step. The authors claim that the methodology is most effective when conducted with a team of requirements engineers with security expertise and stakeholders of the project [1]. SQUARE was developed keeping in mind that most of the systems do not focus on security requirements in the early stages and hence it is made for those systems where security is added as an afterthought and add-ons to the system’s functional requirements [1]. The method considers that a system has only one business goal and multiple security goals derived from that business goal. The method is well-defined and easy to understand and establishes clear and precise goals between stakeholders and developers. There is no mention of asset management and vulnerabilities to the system [2]. The issue of conflicting requirements and validation of the elicited requirements is addressed properly.

A.2 UMLSec

This approach is based on the Unified Modelling Language (UML) for establishing security in critical infrastructure systems [6]. The method focusses on the development of design models, which include security in it along with other features. It aims to help in reducing implementation costs. It extends the already existing UML for the inclusion of security
features and makes use of the fact that UML can be applied to any type of software systems be it object-oriented or not [6]. UMLSec uses different diagrams like Misuse cases, activity diagram, state chart diagrams, sequence diagram, class diagram, deployment diagram and subsystems diagram to model different aspects of the system for security requirements. It introduces three extensions to UML diagrams, stereotypes, tagged values and constraints.

The concept of an adversary is introduced who can harm the system. Threats to a subsystem introduced by an adversary are also modelled. Threats like delete, read, insert and access can be modelled. Important security properties like secrecy, integrity, authenticity, secure information flow, freshness are defined using the UMLSec notations.

UMLSec was developed for the security and safety of critical infrastructure systems in mind. It aims to create a secure environment for the interconnection and smooth functioning of the security-critical systems. The methodology analyses the system at a fairly low level and is suitable for an operational analysis [5]. Most of the security issues are addressed. However the methodology does not focus on elicitation of security requirements, completeness and validation of functional, non-functional and security requirements [2].

A.3 Secure Tropos

Secure Tropos extends the Tropos methodology which enables developers to consider all the stages of software methodology. The paper defines security requirements as a manifestation of a high level organisational policy into the detailed requirements of a specific system. It claims that adding security as an afterthought leads to problems as the requirements have to be fitted into the existing system and may create new problems.

Tropos adopts the i* modelling framework which introduces the notion of actors, goals, tasks, resources and soft goals. The notion of dependency between two actors is introduced to model the diagrams. The depending actor is called depender and the actor on whom the other actor depends for the fulfilment of goal or task is called the dependee. The methodology covers five stages namely, early and late requirements, architectural design, detailed design and deployment. The concept of constraints is introduced and actors, goals, tasks, resources and soft goals are also extended with security in mind [5]. Security constraint, Secure Dependency and Secure stereotypes, tagged values and constraints.

The concept of positive and negative contribution link is added. The positive contribution is added when a node supports the other and negative is added when a node denies the other. All these concepts are added as security concepts in each of the five phases of the development process mentioned earlier.

Secure Tropos is a well-defined process which allows the developers to consider security issues at every stage of system development [5]. This method has an advantage that it can be applied to all the stages of software development. Attacker and threat analysis is possible with the help of the diagrams [2]. Models can be validated based on which one deals with the attacks in the best possible way [2]. Risk assessment and assets are not mentioned specifically.

A.4 CORAS

The authors define CORAS as the security risk analysis method. It uses Unified Modelling Language (UML) for modelling the target system. The CORAS method provides a computerised tool designed to support documenting, maintaining and reporting analysis results through risk modelling, table-based documentation, consistency checking and more [8]. It consists of seven detailed and well-defined steps. CORAS does not focus specifically on security requirements but using the risk assessment and treatment, the developers can understand what security requirements may be needed for the risk treatment. Every step defines who should be involved in the step and what the outcome of the step is. The seven steps are explained below:

1. **Introductory meeting**

   The long-standing credo of requirements engineering reads: ‘‘If you don’t know what you want, it’s hard to do it right.’’[2]. The first step in COARS tries to achieve that. The introductory meeting tries to create a common ground between the stakeholders and the system analysts. The clients present their goals and target of the analysis, what the assets are, what is the scope of the system and what terminologies are to be used.

2. **High level Analysis**

   In this step the analysts produce their side of the system (Target). They present the concepts they understood about the target. The clients and the analysts together identify the assets that need protection as well as the high level analysis of the system. This step produces and asset diagram and a UML diagram for the target description.

3. **Approval**

   This step is the final step of the preparatory stage. Both sides decide on the asset and rank it based on their importance. The consequence and likelihood values for the asset is also defined. A consequence value determines what level of damage can be done to the assets. The likelihood values decides how likely is it to cause damage to the assets. It is very important for all the stakeholders to be present in this meeting because this step sets the stage for the next four steps.

4. **Risk identification**

   This step starts with “Structured Brainstorming” [8]. In this step the stakeholders, all with different backgrounds and level of experience, decide what the risks to the target are. The risks identified are documented and a risk evaluation matrix is created. The final product is the threat diagram with identified threats, vulnerabilities and threat scenarios.
5. **Risk estimation**

In this step, all the threat diagrams are extended with likelihood and consequence estimates. The likelihood estimate decide how much damage can be caused to the asset if the threat is realised. The consequence estimate relates the asset to an unwanted incident. To achieve this, every stakeholder puts forward a possible threat scenario. For those scenarios that are difficult to estimate, help of the analysis leader is taken. This step should involve the technical experts, users and experienced stakeholders.

6. **Risk evaluation**

In this step, the extended threat scenario is used to put all the risks into a risk matrix which helps in deciding which risk should be treated and which is acceptable. After this process, the risk matrix is presented to the client for inspection. A risk diagram containing all possible risks that may need treatment is produced at the end.

7. **Risk treatment**

In the last step of this process, the risks that cannot be accepted into the system are evaluated and countermeasures for the risk reduction are decided. Also, the economic feasibility of the countermeasures is estimated. According to the economic feasibility the countermeasures are finalised. The end result is a treatment and treatment overview diagram.

CORAS however does not focus on elicitation and validation of security requirements at all.

### B. Security of SCADA systems

SCADA systems are widely used for industrial processes such as electric power production and distribution, water treatment and supply, gas and oil production and distribution, telecommunications, chemistry, nuclear fusion [4], [10]. Kang, Lee, Kim and Hur [11] describe it as a twin package of software and network infrastructure into a supervision system that is used to acquire data, analyze it and send control actions.

These are not full control systems but they focus on the supervisory level [10]. Earlier these systems communicated using serial networks and complied with normal standards. The protocols supported minimal functionality and data was sent without encryption or authentication [4]. This is a surprising thing as these systems supported critical infrastructures. This shows that there was little concern for the security of data sent through SCADA systems during that time. However, these days SCADA systems are connected to the internet and are no longer isolated. SCADA systems have become more or less like an enterprise software system as can be seen in Fig. 2.1 [4].

Earlier the SCADA security issues were primarily about physical security but nowadays the security concerns are more about network and communication infrastructures [12]. The integration of SCADA systems with IT systems is also under way which further increases the security risks to these systems [11]. There is a need for proper security measures to be put in place for the protection of these systems. Putting all the security measures at once may not be possible and may hinder the proper functioning of such systems but slowly increasing security measures is a necessity. Too much of security measures also should not be applied, as that may slow down the exchange of data between terminals.

The first step in increasing the security of these systems should be to formulate clear and precise security requirements for the system. In this project I will create a methodology that will try to minimise the security risks to SCADA electricity supply systems by identifying risks and vulnerabilities to those systems and creating a set of security requirements based on those risks and vulnerabilities. There are various ways to assess and reduce risks to SCADA systems. Authors Ralston, Graham and Hieb [4], list out tools and researches published on SCADA risk assessment. Some of the tools mentioned are OCTAVE and CORAS [8]. I have chosen CORAS as the risk assessment method in my methodology because it focusses more on security requirements and defines clearly what methods are being used for security assessment [4].

### III. METHODOLOGY

#### A. Comparison of the methodologies

In this section I will present a comparison between the four methodologies I reviewed. The comparison is based on specific parameters. Those parameters were decided based on their relevance and applicability to the Supervisory Control and Data Acquisition systems (SCADA). The parameters on which the methodologies were evaluated are:

- **Criteria 1**: Are the security triad Confidentiality, Integrity and Availability covered?
- **Criteria 2**: Is the methodology applicable on a system, machine or both?
- **Criteria 3**: Are the views of stakeholders considered?
- **Criteria 4**: Is the concept of asset identification applied?
- **Criteria 5**: Which stage of software development does the methodology apply to?
Table 3.1 shows the comparison of the methodologies.

All the methods have some positive and negative points. Methods like SQUARE are very general and can apply to any system. Also you can use your own methods for risk assessment, requirements elicitation, requirement categorisation and prioritisation. It recommends some methods for each of the processes except risk assessment. The concept of asset identification is left upon the risk assessment method chosen by the stakeholders. It only applies to requirements phase.

UMLSec mainly applies to design phase of the systems and can be used to identify various attack scenarios and vulnerabilities to the system being developed. There is not much emphasis on how the stakeholders will resolve conflicts and validate the requirements. Secure Tropos applies to all stages of software development and considers actors, goals, resources and tasks for modelling. This is particularly useful for systems like SCADA which can be modelled in a similar way. CORAS can be applied to SCADA systems because it provides an automated tool and clearly defines the type of risk analysis like fault tree analysis, vulnerability tree, attack trees to model risks. The concept of asset identification is included in the methodology.

All these methodologies can be combined together to create an efficient and effective methodology.

The methodology aims to bridge the gap between the clients, system analysts and developers of the system or product and to apply it on SCADA electricity supply systems and smart grids. It will set the security requirements straight between the parties and remove any conflicts regarding requirements among them. This will prevent any future project failures and losses.

### B. Proposed Security Requirements Methodology (MAR(S)2)

This section introduces a security requirement methodology that incorporates the features of all the four methodologies and performs all the important functions expected from a well-defined and effective methodology. Figure 3.1 shows the four phases of the methodology and the final artefact that will be produced by the methodology.

The methodology takes four phases to complete. In the first phase, the stakeholders will conduct a meeting and decide upon the definitions of basic and important terms of methodology. The clients, analyst and developers are the stakeholders. The terminologies and definitions can be decided by choosing well-established definitions from standard documents such as IEEE or ISO/IEC. After this step, the business goals of the system are identified. Business goal is something that when achieved will fulfil a need of the organisation in terms of their business. All the stakeholders must be present for this phase. The business goals documented can be one or multiple. The aim of the first phase will clear all ambiguities and reduce the differences between the stakeholders.

In the next phase, the assets of the system are identified. An asset register is created. The main actors and resources in the system are also documented and explained. This concept has been borrowed from the Tropos methodology, which is a goal-oriented approach [7]. Based on the business goals, assets, resources and actors, the security goals are identified. This completes the second phase. The second phase tries to create a clear picture of the system to be built.

In the third phase, based on the actors, security goals and resources the analysts and security experts will create UML diagrams, which have security extensions. UMLSec is recommended here for its specific details and applicability on critical infrastructure systems like SCADA. It produces detailed diagrams, which depicts possible threat scenarios and attacks. After the diagrams are complete, a risk assessment will be completed which will list all the possible risks to the system and how can these risks be treated. This will be accomplished by creating a threat diagram, risk diagram and treatment diagram as in CORAS [8]. This will end the third phase. This phase will make the stakeholders clear about the
system functionalities and security issues. They can also discuss about the possible treatment measures for each security threat.

In the fourth phase, the security requirements will be elicited based on the risk assessment results and UML diagrams. The first set of security requirements’ document is created by the system analysts and developers. The document is validated and the completeness of the requirements is checked by discussing it with the client. After validation and resolution of the conflicts the final product is generated. The final product as shown in the figure is a security requirements document.

C. Advantages of Security Requirements Methodology (MAR(S)2)

The proposed methodology includes important steps in a security requirements engineering process like asset management, security design and diagrams, risk assessment, requirements elicitation and requirements validation. Most of the current methodologies lack one or few of the steps, which leaves a space for security threats to sprawl in. This methodology is also generic in a way and can be applied to any software system, including SCADA systems. The feasibility of the methodology relies on the fact that all the steps are easy to understand and are clearly defined. It incorporates the ideology of many popular methods such as CORAS and UMLSec, which are already in use by many organisations and industries. This will make the methodology easy to fit in the existing scenario. But none of these methodologies were specifically designed for SCADA systems, hence the methodology of MAR(S)2 is designed with consideration to the architecture of SCADA.

IV. Future Scope and Implementation

This methodology is suited for future needs as it takes into account the major security concerns related to critical infrastructures like SCADA. It can be used to reduce and mitigate major threats and concerns in such systems. This can be tested by the help of surveys and actual implementation in the SCADA industry. The major challenge for this methodology is to see how the organisations adopt it into their working culture and practices. The methodology can act as a middle-man between the SCADA system clients and suppliers. Phases 1, 2 and 4 of the methodology can be implemented either through face-to-face discussion and meetings or by using an automated software as they require interaction among clients and suppliers of the system. The third phase however requires only the security team to complete. Tools available for making UML diagrams with security extensions such as UMLSec can be used. For the risk assessment, methods such as CORAS can be used. Thus the implementation of the methodology is also quite easy and can be incorporated into the current working practices of the organizations.

CONCLUSION

MAR(S)2 strives to reduce the security threats and vulnerabilities in the SCADA systems. It addresses all the major security concerns like asset management, threats and risk assessment and solves major requirements problems like validation and conflicts. This methodology can be applied in the SCADA industry for documenting clear, precise and validated security requirements. This can solve major security problems in the SCADA industry and can be helpful in making these systems secure against major threats which is the need of the hour.

REFERENCES