Mälardalen University Licentiate Thesis 296

Access Control Models to secure Industry 4.0 Industrial Automation and Control Systems

Björn Leander

MÄLARDALEN UNIVERSITY
SWEDEN

Mälardalen University Press Licentiate Theses
No. 296

ACCESS CONTROL FOR SECURE INDUSTRY 4.0 INDUSTRIAL AUTOMATION AND CONTROL SYSTEMS

Björn Leander

2020

School of Innovation, Design and Engineering

ISBN 978-91-7485-478-7
ISSN 1651-9256
Printed by E-Print AB, Stockholm, Sweden

Abstract

A significant part of our daily lives is dependent on the continuous operation of Industrial Automation and Control Systems (IACS). They are used to control the processes of delivering electricity and clean water to our households, to run and supervise manufacturing industries that produce things we use every day. Therefore, undisturbed, safe and secure operation of IACS are highly important for us all. A malfunctioning IACS may cause damage to the environment, stop production of goods or disrupt essential infrastructure.

The ongoing transformations related to the Industry 4.0 paradigm is having a great impact on IACS, forcing a shift from a rigid, hard-wired system architecture towards a service-oriented structure, where different modules can collaborate dynamically to adapt to volatile production requirements. This shift entails a substantial increase in connectivity and is hence potentially increasing exposure of these systems to cybersecurity threats. Understanding potential risks, and protection against such threats are of great importance.

Access Control is one of the main security mechanisms in a software system, aiming at limiting access to resources to privileged entities. Within IACS, this mechanism is mainly used as means to limit human users' privileges on system assets. In the dynamic manufacturing systems of Industry 4.0, there is a need to include fine-grained Access Control also between devices, raising a number of issues with regards to policy formulation and management.

This licentiate thesis contributes towards the overall goal of improving the security of IACS in the evolving systems of Industry 4.0 by (1) discussing high-level security challenges of large industrial IoT systems, (2) assess one of the main standards for IACS cybersecurity from an Industry 4.0 perspective, (3) derive requirements on Access Control models within a smart manufacturing system, and (4) presenting an algorithm for automatic Access Control policy generation within the context of modular automation, based on formal process descriptions.

Sammanfattning

En stor del av vår vardag är beroende av att Industriella automations- och reglersystem (IACS, Industrial Automation and Control System) fungerar problemfritt. Sådana system används för att leverera elektricitet och rent vatten till våra hem, och till att tillverka produkter vi använder varje dag. Därför är säker drift av IACS en nödvändig samhällsnytta. En felaktig IACS kan leda till skador på miljö eller människor, hindra produktion av livsmedel, m.m.

Industri 4.0 innebär en förändring inom tillverkningsindustrin, med påverkan på många befintliga och framtida IACS. Detta teknikskifte leder bl.a. till att den statiska miljö som finns i traditionella produktionssystem kommer ersättas av sammankopplade dynamiska system som momentant anpassas efter behov. Detta förändrade beteende leder till nya risker relaterade till cybersäkerhet. Förståelse för dessa risker är av stor vikt för att bibehålla säker drift av framtidens industriella automationssystem.

Åtkomstkontroll är en viktig säkerhetsmekanism i ett mjukvarusystem, som används för att begränsa åtkomst till systemets resurser. Inom industriella reglersystem har åtkomstkontroll främst använts för att begränsa människors rättigheter att utföra operationer på tekniska komponenter. Inom Industri 4.0 finns behov av detaljerad åtkomstkontroll även mellan komponenterna i systemet, vilket leder till en mängd problem relaterat till hur regler för åtkomstkontroll ska formuleras och upprätthållas.

Denna licentiatavhandling bidrar till att förbättra säkerheten för IACS i relation till det pågående teknikskiftet inom Industri 4.0 genom att (1) diskutera utmaningar relaterade till cybersäkerhet för dessa system, (2) utvärdera en av de viktigaste industriella standarderna i relation till Industri 4.0, (3) formulera krav på modeller för åtkomstkontroll, och (4) presentera en algoritm som automatiskt formulerar regler för åtkomstkontroll inom modulär automation, utgående ifrån formella processbeskrivningar.

Acknowledgments

First of all I want to express my gratitude to my supervisor team, Prof. Hans Hansson and Dr. Aida Čaušević at Mälardalen University and Tomas Lindström at ABB Industrial Automation. Thank you for time, guidance and confidence in me. Without that this thesis would not have been possible!

I would like to thank all the colleagues and managers supporting my work at ABB IA, especially to Jonas Stigeberg and Martin Andersson for trust and patience when joining the ARRAY program. Thank you Thomas Nolte for founding the ARRAY program, without that initiative I would not be in the academic world.

To my colleagues at Mälardalen University, thank you for being good company and interesting discussion partners at fika and lunch. To my fellow PhD students, thank you for all the time spent discussing research, courses, conferences, work, and life in general.

To my parents Lars and Siv: everything I know starts with you, thanks for your unconditional support and guidance through my life. Lars brought me into the lucrative world of computer science as a bug hunter (monkey testing) when I was seven years old, offering 5kr for each found bug. Little I knew that this was the first step in my career.

Finally my deepest gratitude goes to my family, Linnea - the love of my life, and my children Hugo, Nike and Teo - without you my life would be dull!

This research is supported by ABB Industrial Automation and the industrial postgraduate school Automation Region Research Academy (ARRAY), funded by The Knowledge Foundation.

Björn Leander
Västerås, September 2020

List of Publications

Publications included in thesis[1]

Article A: *Cybersecurity Challenges in Large IIoT Systems*, Björn Leander, Aida Čaušević, Hans Hansson, In the Proceedings of the 24th International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, September 2019

Article B: *Applicability of the IEC 62443 standard in Industry 4.0 / IIoT*, Björn Leander, Aida Čaušević, Hans Hansson, In the Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES), Canterbury, United Kingdom, August 2019

Article C: *Access Control for Smart Manufacturing Systems*, Björn Leander, Aida Čaušević, Hans Hansson, Tomas Lindström, In the proceedings of the 14th European Conference on Software Architecture, 2nd Workshop on Systems, Architectures, and Solutions for Industry 4.0 (SASI4), L'Aquila, Italy, September 2020

Article D: *A Recipe-based Algorithm for Access Control in Modular Automation Systems*, Björn Leander, Aida Čaušević, Hans Hansson, MRTC Report, MDH-MRTC-333/2020-1-SE, Mälardalen Real-Time Research Centre, Mälardalen University, 2020

[1]The included publications have been reformatted to comply with the thesis layout.

Publications not included in thesis

Article E: *Classification of PROFINET I/O Configurations utilizing Neural Networks*, Bjarne Johansson, Björn Leander, Aida Čaušević, Alessandro Papadopoulos, Thomas Nolte, In the Proceedings of the 24th International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, September 2019

Article F: *Towards an Access Control in a Smart Manufacturing Context*, Björn Leander, MRTC Report, ISRN MDH-MRTC-329/2020-1-SE, Mälardalen Real-Time Research Centre, Mälardalen University, 2020

Contents

I Thesis **1**

1 Introduction **3**
1.1 Thesis outline 5

2 Background **7**
2.1 Industrial Control Systems and Industry 4.0 7
2.2 Cybersecurity 13
2.3 Access Control 18

3 Research Summary **25**
3.1 Research Process 25
3.2 Research Goals 26

4 Contributions **29**
4.1 Included articles 30

5 Related Work **33**
5.1 Cybersecurity in Industry 4.0 33
5.2 Dynamic Access Control 34
5.3 Smart and Modular Manufacturing systems 35

6 Conclusions **37**
6.1 Summary of contributions 37
6.2 Future directions 37

II Included Articles **47**

7 Article A:
Cybersecurity Challenges in Large IIoT Systems **49**
7.1 Introduction 51
7.2 Background 52
7.3 A working example 54
7.4 A Threat Model from an IIoT perspective 55
7.5 Challenges and future directions 62
7.6 Related Work 66
7.7 Conclusions 67

8 Article B:
Applicability of the IEC 62443 standard in Industry 4.0 / IIoT **71**
8.1 Introduction 73
8.2 Background 74
8.3 IEC 62443 - Current state 75
8.4 Assessment of IEC 62443 in relation to IIoT 84
8.5 Conclusions 90

9 Article C:
Access Control for Smart Manufacturing Systems **95**
9.1 Introduction 97
9.2 Background 98
9.3 Access Control Requirements on Smart Manufacturing 101
9.4 A Smart manufacturing Scenario 104
9.5 Fulfillment of requirements 106
9.6 Related Work 109
9.7 Conclusions 110

10 Article D:
Recipe Based Access Control in Modular Automation **115**
10.1 Introduction 117
10.2 Preliminaries 119
10.3 Generating Access Control rules in NGAC using an SFC Recipe 125
10.4 Proposed algorithm exemplified 131
10.5 Discussion 132
10.6 Related Work 136
10.7 Conclusions 137

Part I

Thesis

Chapter 1

Introduction

Industrial Automation and Control Systems (IACS) are used for operating a wide range of industrial applications, including critical infrastructure, such as power plants and clean water supplies [69]. Industry 4.0 [37, 19, 43] is a paradigm shift currently shaping the future of IACS, implying huge changes both from a business and technological perspective. The aim of Industry 4.0 is to enable optimization, cost-savings, and new business opportunities in different domains, and it is expected to introduce significant advances in optimizing decision-making, operations and collaborations among a large number of increasingly autonomous control systems [26].

In the dynamic and flexible systems of Industry 4.0, communication paths are often not pre-defined, and production schemes are ever-changing. Therefore it becomes difficult to detect malicious behavior [73], especially between devices seen as legitimate. At the same time, the attack surface and complexity of the systems are increasing, raising the risk of a legitimate device being compromised [77].

A compromised device, controlled by a malicious actor, may cause a significant economic damage for the factory owner, as well physical damage on e.g., humans, machinery and the environment. The impact may be direct, e.g., the opening of a valve may overfill a tank or turning on heating in an empty reactor may cause a fire. Impact could also be indirect, e.g., changing ratios of materials used to produce a medicine may render it harmful. The direct causes are usually mitigated by implementations of secondary safety measures, while indirect causes may be more difficult to detect and mitigate.

During the last years, there has been a steady trend of increasing amounts

of cyber-attacks on industrial control systems [68]. When analyzing who performs attacks against different targets, there are a number of standard categories [56, 32] used: *hobby hacker, insider, cyber-criminal, hacktivist, terrorist and nation state.* For attacks against industrial control systems, the two main categories with knowledge and capacity to perform targeted attacks are the *insider* and the *nation state.* However, any of the other categories can use an *insider* to gain initial foothold, e.g., by social engineering, bribery or extortion. An *insider* can hold deep knowledge of the system, credentials, as well as physical access to the system.

Applying strict and fine-grained Access Control according to the principle of *least-privilege* [62] is one of the major mechanisms used to protect against the threat from insider attacks, by allowing access to operations or data only to privileged entities. It also increases the visibility of the malicious actor, as denied Access Control requests are typically monitored e.g., using a Security Information and Event Monitoring (SIEM) system [24]. However, using a strict Access Control at the lower layers in an automation system is quite uncommon. Historically, industrial automation systems have been built up using proprietary communication protocols, hard-wiring between controllers and IO, and the notion of an air-gapped network, i.e., no communication between the control network and the outside world. These assumptions on the technical solutions have meant that the pragmatic solution used is to allow all legitimate devices on the network to perform any action. With the advent of Industry 4.0, none of these assumptions hold anymore, and therefore the practice of including a strict Access Control between devices in modern IACS is of increasing importance.

Research presented in the thesis aims to analyze and mitigate cybersecurity risks in IACS within Industry 4.0 and Smart Manufacturing by improving Access Control Models. By understanding cybersecurity requirements we can analyze gaps in the state of the art, and suggest improvements. By assessing the available standardization frameworks in the context of Industrial Internet of Things (IIoT) systems, we can provide up to date guidance used by industry and certification agencies. Evaluation of Access Control models within Smart Manufacturing and Modular Automation system, and improvements of these models, can facilitate a wider adaption of manufacturing systems to utilize fine grained Access Control, thereby increasing their resilience against certain cybersecurity threats.

The following contributions are included in this licentiate thesis:

- An analysis of identified gaps in state of the art with regards to cyber-

security in IIoT systems.

- An analysis of cybersecurity requirements on IIoT systems.
- An analysis on how the existing cybersecurity standard IEC 62443 can cater for identified gaps.
- A list of requirements on Access Control models in Smart Manufacturing systems.
- A recipe-based automatic Access Control policy generation algorithm for Modular Automation systems, along with a formal proof validating the algorithm.

1.1 Thesis outline

The thesis is organized in two major parts. In Part I the background, research goals, related work and summarized contribution of the thesis is described. Part II consists of the four articles detailing the work.

The remainder of Part I is organized as follows:

Chapter 2 provides necessary technical background to understand motivation and challenges for the conducted research.

Chapter 3 describes the research process and methodologies used within in our research. The high level motivation of the research and the resulting research goals are detailed.

Chapter 4 summarizes contributions of the included articles, a short summary, and relation to the formulated research goals.

Chapter 5 introduces a relevant related research.

Chapter 6 summarizes thesis contributions, along with suggestions for future studies.

Chapter 2

Background

In the following we present background necessary for understanding the proposed work. The background is divided in three main parts. The first part is describing general background related to industrial control systems and the evolution of the Industry 4.0/Industrial Internet of Things. The second part focuses on challenges related to cybersecurity arising with the Industry 4.0 systems. The last section introduces necessary background related to Access Control, which is one of the main cybersecurity mechanisms studied in this thesis.

2.1 Industrial Control Systems and Industry 4.0

2.1.1 Industrial Automation and Control Systems

Industrial processes are to a large extent automated and supervised by computer systems known as Industrial Automation and Control Systems (IACS). They are used in a large variety of applications such as power production facilities, clean water plants, large ships, process manufacturing, tunnel ventilation, data-center power distribution, etc. The traditional IACS follow the Purdue Enterprise Reference Architecture (PERA) [82], as illustrated in Fig. 2.1a. The goal of an IACS is to provide a cost-efficient and safe operation of a physical process. The process is monitored and controlled through a set of sensors and actuators. A number of Programmable Logical Controllers (PLCs) are connected to the sensors and actuators. Typically, a PLC contains logic to automate a sub-process within the IACS. Above the controllers, a supervisory system exist, where operators control production by altering set-points, han-

dle alarms and events, etc. The production demand and operational planning are based on decisions of the operational management Manufacturing Execution System (MES), where current production data is combined with information and decisions from the high level Enterprise Resource Planning strategies (ERP). The technological solutions used in the lower layers of PERA (Levels 1-2) being directly focused on the operations of the physical process are usually named as Operations Technology (OT), while Levels 3-4 contain ubiquitous components used within standard Information Technology (IT) environments. IT and OT networks contain different kinds of components, and are physically separated, to ensure safe and secure operations [24].

This architecture of controlling industrial processes can be seen as the result of the 3^{rd} industrial revolution, where electronics has been introduced in production environments, enabling automatic closed loop control (second half of the 20th century). The previous industrial revolutions encompass first the inclusion of steam and water powered machinery (late 18th, early 19th century), and second the electrification of the manufacturing process (late 19th century) [54].

The market for component and system development for IACS are dominated by a relatively small number of large companies, e.g., ABB, Emerson, Rockwell, Schneider Electrics, Siemens, etc. Systems and components have historically used proprietary technologies and protocols, however, customer demands have forced solutions for hybrid systems, comprising e.g., controllers from several vendors, being supervised through a Distributed Control System (DCS) from yet another vendor.

2.1.2 Industry 4.0 and the Industrial Internet of Things (IIoT)

The 4^{th} industrial revolution is expected to occur during the early 21st century, driven by an accumulated body of innovations in the area of information technology, Internet of Things (IoT), Artificial Intelligence, big-data, etc [54]. The concept of Industry 4.0 was first introduced by the German government in 2011, as a program to increase competitiveness of domestic manufacturing industry [71, 75]. Similar initiatives have been taken in other parts of the world, e.g., the Industrial Internet Coalition (IIC) and Smart Manufacturing Leadership Coalition (SMLC) in North America.

These initiatives have resulted in a number of standards and reference architectures e.g., Reference Architecture Module for Industry 4.0 (RAMI4.0) [23] suggested by the International Electrotechnical Commission (IEC), and the IIoT Infrastructure [27] suggested by the IIC. More details on these reference

architectures for Internet of Things (IoT) is provided in a survey by Weyrich et al. [80].

The technical systems of Industry 4.0, transforms IACS from following the strict hierarchical structure, as described in PERA, towards a mesh-like self organizing structure [65, 44], as depicted in Figure 2.1b. In this aspect, Industry 4.0 is accelerating an already on-going trend towards a convergence between the IT and OT [33]. Industry 4.0 also introduces the concept of IoT and Services into the industrial domain, e.g., using light-weight smart sensors for collecting and distributing process data, and cloud services for access to the data, as well as inference aid to decision makers. IIoT is however often seen as encompassing a wider area than the mere industrial applications, including e.g., smart cities [51], smart healthcare [5], intelligent transportation systems [39], etc. To be precise, we can say that the Industry 4.0 concept encompass a holistic view of the whole manufacturing and process industries, including novel business models, inter-organization cooperation, logistics, process system, etc. The IIoT on the other hand is a technological domain where Cyber-Phyiscal Systems (CPS) [4], IoT and the internet of services are integrated into industrial applications. Industry 4.0 uses technical systems based on IIoT to address some of its requirements.

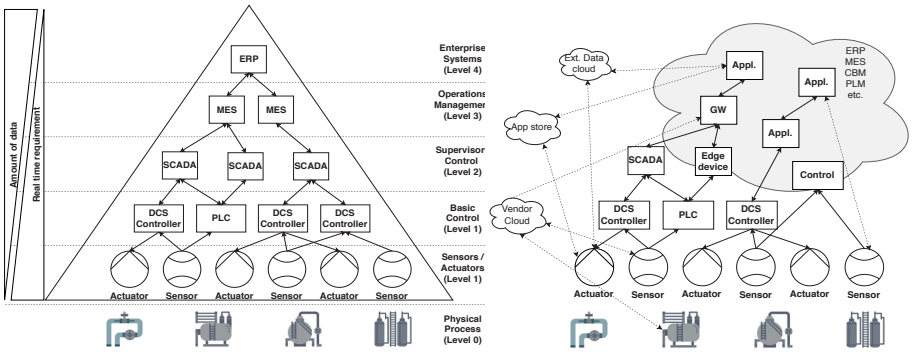
Several companies are working on developing technology related to Industry 4.0. Among the traditional providers in control systems emphasis is typically put on integrating brownfield control-system installations into cloud solutions, by e.g., adding data concentrators that can publish control system data to the cloud. Some examples of existing solutions include: the Industrial Edge[1] from Siemens and ABBs IA Edge[2] and Yokogawa[3]. For all of these industrial initiatives, cybersecurity is seen as an important challenge, and there is a lot of effort being put into ensuring that transfer of data from control system to cloud can be achieved in a secure way, and that the data is protected once placed in the cloud.

With the emerging Industrial Internet, companies that traditionally are working with open or general purpose systems are becoming increasingly important. For example, Microsoft, Amazon and Google, are providing cloud solutions to be utilized in IIoT-systems; Cisco, Westermo and other players within the network and switches community are designing and implementing key functionality for virtual network segmentations, Software Defined Networks,

[1] `new.siemens.com/global/en/products/automation/topic-areas/industrial-edge.html`

[2] `new.abb.com/abb-ability/`

[3] `www.yokogawa.com/library/resources/white-papers/dx-arc-wp/`



(a) Automation Triangle **(b)** IIoT architecture example

Figure 2.1: Traditional IACS and IIoT Architectures

etc; and Ericsson and other companies within the telecom-industry focus on providing solutions for communication, 5G technology being one of the enabling technologies for Industry 4.0.

2.1.3 Smart Manufacturing and Modular Automation

Smart manufacturing [47, 9] and Modular Automation [35] can be seen as evolving technologies of the Industry 4.0 paradigm, within the manufacturing industry domain. Smart Manufacturing encompass discrete production, while Modular Automation encompass the continuous manufacturing, e.g., chemical, energy and pharmaceutical industries.

Industry 4.0 as a whole, and these specific domains, share a number of developing trends, driving a lot of current academic and industrial efforts towards related technical solutions. The aim of these trends is to enable optimization, cost-savings, and new business opportunities in different domains, and significant advances are expected in optimizing decision-making, operations and collaborations among a large number of increasingly autonomous control systems [26].

One of the most important achievements in these domains is customer-oriented production, where the current customer demands and requirements directly have impact on what and when to start the production. Drawn to its furthest, this requires manufacturers to be able to support what is called a *mass customization* [43], meaning that every produced unit is tailor-fit based on a specific customer demand. This is a far departure from the traditional manufacturing environment, where a lot of effort is first spent in developing

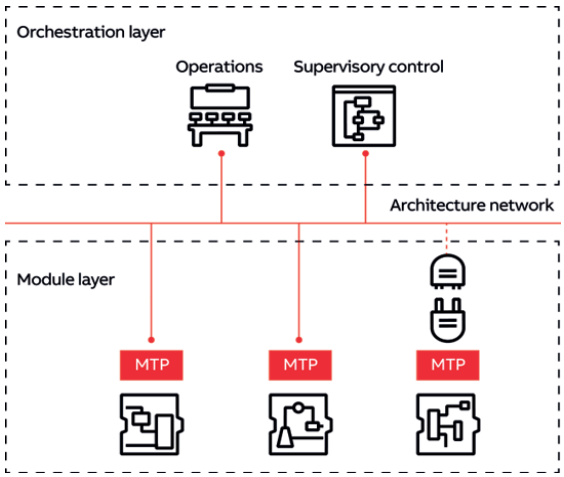


Figure 2.2: Modular Automation MTP-Architecture

a product and a production line being able to manufacture large volumes of identical units of that product. To achieve such a dynamic behavior in a manufacturing environment, while staying economically competitive, the production system is defined as a collection of modules able to complete specific manufacturing tasks. To create a customized product, these modules are combined and configured in a specific way in order to fulfill the customer requirements. In modular automation, modules providing specific functionality are described as Module Type Packages (MTP), that are used when formulating recipes. Specific modules being instances of an MTP are then used in production, based on an orchestration architecture (see Figure 2.2[4]).

Another related trend within Industry 4.0 is to shorten the feedback loop between high-level enterprise decisions and a low-level production [9]. Using a traditional system architecture for IACS, there may be a considerable amount of time from a detected issue on the field level to a enterprise level decision of change to implementation in the manufacturing system [44]. A shorter lead-time will reduce costs and make the overall system more agile and allow fast adaptations to high-level market demands. Technically, this is supported by making data from the production environment available for data analysts working with cloud-solutions and delivering aggregated information to decision makers, usually utilizing Artificial Intelligence (AI) for inference. Often,

[4]Image source from ABB: `new.abb.com/control-systems/modular-automation/module-type-package`

the sensor networks used in industrial settings and publishing or concentrating data to cloud or fog solutions are described as being a subset of the IIoT.

A third related trend, also seen in society as a whole, is focused towards a higher degree of *servitization* [71, 74]. Servitization means that instead of buying a product that will fulfill a specific task, one buys the service of the task when needed. For cloud computing this business model is already the dominant one [76]. For IACS it is suggested that the autonomous modules discussed earlier will be provided as a service by companies that are specialized in the specific tasks the module should perform. For example, this could mean that a manufacturing industry could buy the service of packaging as a service from a company specialized in building packaging robots. The company offering the packaging service must be able to monitor, service and replace equipment in order to promise a certain quality of service. For a whole manufacturing environment this indicates a vast amount of new stakeholders in need of direct access to their respective part of the system. There are also efforts in the direction of servitization related to the whole manufacturing process, in which design as a service, manufacturing as a service, logistics as a service, etc., would be combined [34].

2.1.4 The Open Platform Communication Unified Architecture (OPC UA)

One of the main challenges within IIoT systems is how to reach interoperability between a potentially diverse set of heterogeneous devices that must be able to interact in order for the system to fulfill its tasks. The Open Platform Communication Unified Architecture (OPC UA) [25] is a communication protocol used for inter-machine communication in industrial control systems, and is of increasing interest for use in modern automation systems. It is a protocol based on a Service Oriented Architecture (SOA), typically running on TCP/IP networks. OPC UA is currently the main candidate for providing interoperable communication between entities within IIoT systems [35, 78, 44]. Several organizations are including so-called *companion specifications* into their standards, describing how OPC UA should be implemented to reach compliance, e.g., with regards to security services. The Open Process Automation Standard 1.0[5] (OPAS 1.0) defined by Open Process Automation Forum (OPAF) is one example of a standard containing such a companion specification. OPC UA is able to allow interoperability at protocol level, but to reach semantic interoperability, there must be additional mechanisms, for example by using

[5] `https://publications.opengroup.org/c19f`

AutomationML as a basis, provided by Henßen et al. [18].

2.2 Cybersecurity

Cybersecurity is the protection of a computer system from unauthorized actors' possibility to: (1) steal or alter information in the system, (2) disrupt or alter behavior of a function or (3) perform an unauthorized action [29]. Selecting the cybersecurity protection mechanisms for a system, requires a trade-off between cost, usability and security. A mechanism may, e.g., be too labor-intensive to justify in relation to the value of the asset it protects, another mechanism may limit the system availability so that the system cannot fulfill its intended function. Such characteristics transforms cybersecurity into risk management [20].

To evaluate and mitigate risk with regards to cybersecurity, there are several methods that can be used as an aid. One commonly used method is Threat Modeling [50], in which the system is modeled, usually in a data flow diagram, and all component interactions are systematically evaluated to list potential threats to the system. These threats can then be evaluated (e.g., using the Common Vulnerability Scoring System [14]) and mitigated or removed, and the residual threat evaluated. In this way different mitigation strategies can be selected, requirements on cybersecurity can be elicited, and the overall residual risk for cybersecurity related incidents can be evaluated.

Another aspect of cybersecurity is building a sufficient level of trust or dependability that can be put into the system. As described by Madsen [45], the trustworthiness of an information system is the degree of confidence that it performs as expected with respect to key characteristics during unexpected scenarios, such as: disruptions from the environment, human errors, system faults, and attacks from adversaries.

The CIA-model is often used to describe the desired security characteristics of a system. CIA stands for **C**onfidentiality, **I**ntegrity and **A**vailability [81]. Confidentiality is the characteristic protecting against unintended disclosure of information, typically provided by encryption and authorization. Integrity ensures that data cannot be altered without detection [79], typically provided by cryptographic hash-sums and a signature. The availability relates to keeping the system running despite different types of disruptions. Methods for protecting the availability of a system includes, e.g, firewalls, anti-malware software, network segmentation, intrusion prevention systems, etc. In IT-systems the importance of these characteristics are typically weighed in order

of appearance, i.e., confidentiality is valued higher than availability [33].

2.2.1 IACS and Cybersecurity

In IACS, cybersecurity protection is part of the overall goal of ensuring safe and secure operations of the physical process, against negative Health, Safety and Environmental (HSE) impact [24, 33]. Whereas CIA is the norm in IT systems, some argue that for industrial systems it is Safety, Reliability and Availability that are the guiding principles [38]. This indicates that traditional cybersecurity measures may not fit the solutions of IACS. For example, one common mitigation strategy in a compromised IT-system is to simply turn off the implicated component. Such a strategy may not be feasible in a running production system, as halting production equipment could have dire economical as well as environmental consequences [10].

In IT systems, as well as within IACS, important strategies for cybersecurity includes:

- Segmentation [24]: Divide the network into zones based on criticality, and add perimeter protection between zones (e.g., firewalls).
- Defence-in-depth [1]: There is not one single mechanism that will handle all possible threats, instead using a layered approach with several complementing mechanisms provides an overall system security.
- Built-in security or Secure by design [21]: Cybersecurity shall be an intrinsic part of the component and system development process, rather than functions added on top of an existing system.

Several of the companies in the cybersecurity business (e.g., F-Secure, FireEye, Kaspersky, etc.) also provide solutions in the area of cybersecurity for industrial systems. To some extent these efforts are along the way of using traditional cybersecurity solutions from the IT world applied to the IACS, e.g., applying anti-malware or intrusion detection mechanisms. There are however also solutions more specifically tailored towards OT security, such as the Tenable.ot product.

Few companies are working specifically with Access Control. One exception is Object Security, grounded in research regarding Model Driven Security and have products for automated policy generation from domain specific models. The focus of their work does however seem to be mainly on traditional IT-systems, and for rather static models, compared to the dynamically changing models of e.g., Modular Automation.

2.2.2 Standardization

When developing, deploying and operating IACS, standardization plays an important role for utilization of cybersecurity mechanisms [10, 53]. For some applications, a process owner is required to follow a specific cybersecurity standard (e.g., NERC CIP[6]), system developers may be obliged to fulfill specific certifications (e.g., SDLA[7], EDSA[8], Common Criteria[9], etc.), usually prescribed by industrial standards. In any case, the standards are what IACS are measured against. IEC 62443 [24] is one of the most used cybersecurity standards for industrial control systems [33]. An IACS owner can use the methods and requirements described in IEC 62443 to keep its system at a desired level of security. Moreover, the IACS owner in most cases require that service providers and manufacturers of the components used in the IACS follow the principles and adhere to a certain security level of the standard for their delivery. In this way the IEC 62443 is a source of common understanding of cybersecurity related issues for IACS owners, component developers, and service providers. Standardization frameworks are usually developed over a long period of time, and there is an apparent risk that they are outdated during quick technological shifts, as the one related to Industry 4.0.

2.2.3 Motivations behind attacking an IACS

During the last years, there has been a steady trend of increasing amounts of cyber-attacks on IACS [68]. When analyzing who and why attacks occur against different targets, a number of standard categories [56, 32] can be identified, see Table. 2.1.

ID	Category	Motivation	Capability
1	Basic user	hobby, show-off, etc.	Low
2	Insider	economical, personal, tricked	Low-High
3	Cyber-criminal	economical	Low-Medium
4	Hacktivist	visibility, political, sowing distrust, etc.	Low-Medium
5	Terrorist	visibility, causing damage	Low-Medium
6	Nation state	espionage, military / defense	High

Table 2.1: Attacker categorizations, synthesized from [56, 73]

[6]North American Electric Reliability Corporation, Critical Infrastructure Protection
[7]ISASecure Certification - Secure Development Lifecycle Assessment
[8]Embedded Device Security Assessment
[9]Common Criteria for Information Technology Security Evaluation ISO/IEC 15408

Attack attribution is a difficult subject within cybersecurity, the most skillfully executed attacks may never be exposed, attackers will use their skills to hide or obfuscate the origin of the attack, making attribution difficult. It is, however, not uncommon that an attack is attributed to a specific hacker group after forensic analysis, the group is often loosely related to e.g., a criminal network or a national state.

Considering the different motives for the categories of attackers, currently the economical benefits of attacking an IACS are not high enough - using ransomware for extortion or similar types of an attack that may motivate a cybercriminal are easier to distribute on a large scale towards targets within traditional IT-environments, see for example the NotPetya [17] or WannaCry [48] attacks. The same rational would make the Basic user category turn their attention towards easier targets. This remains true as long as the IACS are air-gapped and built upon specific purpose equipment, protocols, etc. In the future, both these categories may start to target IACS, and IACS components built upon IT technologies may be collaterally affected by attacks with a wide scope.

For a hacktivist organization, attacking an IACS may be an interesting target, halting or threatening to halt a critical infrastructure would surely lead to high exposure. Similarly, for the terrorist category, attacking a critical infrastructure or an important manufacturing industry can cause damage on a military scale. Currently both these categories would probably find worthy targets within the IACS segment, but the effort is here possibly too high to be able to perform the very sophisticated types of attacks required to reach the desired goals.

The nation state sponsored attacks are currently the ones posing the most acute threats to IACS [70]. For a nation-state there may be a great military and political advantage in having access to a critical infrastructure and similar facilities of national interest (e.g., financial or communication) for a potential adversary, for reconnaissance, intelligence and as a potential support at military operations. The cost of launching the attack is relatively low, compared to conventional espionage and military operations.

The insider can be anyone within the IACS organization that have the motivation to perform an attack. It may be a disgruntled employee or contractor (e.g., as in the Maroochy incident [67]), an engineer or operator tricked by a social engineering attack, or someone recruited/bribed by any of the categories 3-6 organizations described in Table 2.1. The insider may have very deep knowledge of the system, and typically holds credentials and authoriza-

tion data to perform very sophisticated and targeted attacks. Insider attacks can e.g., be launched form of a rogue device placed on the control network, or a malware/backdoor installed on a legitimate device within the IACS.

The combination of *insider* and *terrorist* or *nation state* sponsored attacks seems to be the most dangerous and potent combination. The STUXNET is one example of such a state-sponsored operation, were an IACS employee has been tricked into plugging in a USB-memory stick into its computer, unknowingly infecting it for further lateral movement throughout the system, until it reached the intended target, in this case being an Uranium concentrating centrifuge in an Iranian state-owned laboratory. This has led to destroying the laboratory equipment, thus effectively slowing the nuclear weapon capabilities of Iran [12].

2.2.4 Challenges related to IIoT/Industry 4.0

As a result of the technological developments related to the Industry 4.0 paradigm, the industrial control systems of the future have a different set of characteristics compared to a traditional IACS. Figure 2.3 summarize different categories of entities that exist and interact in an IIoT system, and examples of their respective vulnerabilities. With regards to cybersecurity, the implications are, for example:

- Drastically increased attack surface, due to interconnections between different devices, systems and the outside world [77].
- Increased flexibility and dynamicity leading to a difficulty in detecting anomalies in system behavior [73].
- New groups of stakeholders and users with access to data and functionality within the systems [44, 34], also increasing the social attack-surface.
- Increasing amount of devices and software within the IACS, increasing the management effort for keeping devices up to date, etc. [49].

These implications incite a wider range of attackers, increasing the economical potential and decreasing the required effort of an attack. Countering these cybersecurity challenges related to the emerging characteristics of Industry 4.0 systems are therefore of great importance.

For a cybersecurity attack against an IACS to have impact on the controlled process, the attacker must be able to move laterally between less protected (where initial foothold is gained) to higher protected zones [10]. The consequences of Industry 4.0 is that the number of potential paths an attacker can

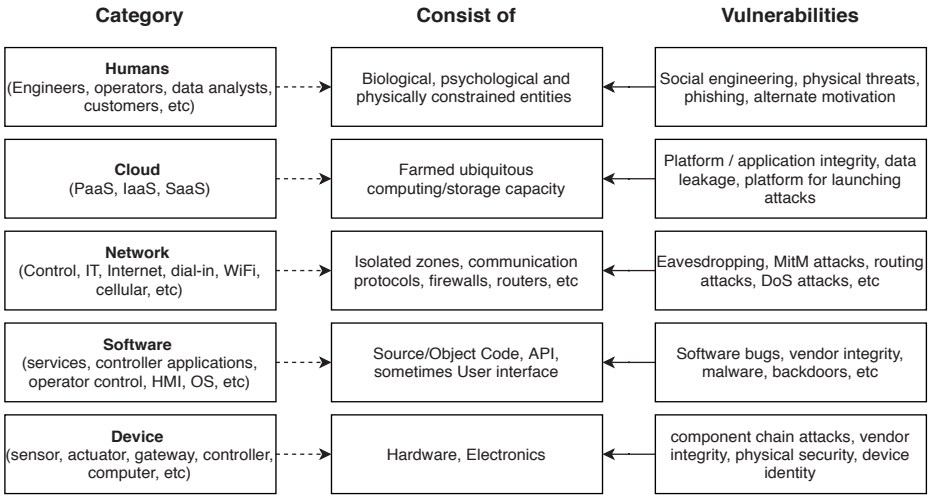


Figure 2.3: Entities and related attack surface in an IIoT system

take to reach its intended target has increased.

Against lateral movement between zones in the control network, and for executing operations in the system, there are a number of primary protective mechanisms: Authentication, Access Control and perimeter protection. Authentication will disallow entities without valid credentials to operate within the system. Access Control will limit privileges for an authenticated entity according to a set of policies. Perimeter protection, such as firewalls, Intrusion Detection and Protection systems (IDS/IPS), impede movement cross network zones. Authentication and different perimeter protection mechanisms are quite widely used within IACS today. The usage of fine-grained Access Control is however not very mature within IACS, due to the effort and complexity needed to achieve it [32]. Furthermore, fine grained Access Control may be the best protection against the *insider* attacker category, where need for lateral movement of the attacker may be small, and credentials are already compromised. Due to its importance, Access Control is the main topic of this thesis, described more in detail in the following section.

2.3 Access Control

Access Control is the mechanism granting or denying a request from a subject to access a resource [64]. Other terms used with equal or similar meaning as Access Control are Privilege Handling and Authorization. As Access Control

in most cases is related to connecting a specific subject to privileges on a specific object, secure identification of entities (authentication) is a prerequisite for effective Access Control. An attack breaching the intentions of a formulated Access Control policy is called an *elevation of privilege* [46] or *privilege escalation attack*, with the implication that a malicious actor is attempting to gain access to privileges or resources other than the intention of the policy.

Apart from limiting privileges according to formulated policies, Access Control is also used for detection of attempted privilege escalation attacks, or forensic analysis after a confirmed attack, as it is a common practice to keep and monitor audit logs for security events related to Access Control, e.g., according to IEC 62443 [24].

2.3.1 Principles

Sandhu el al. [63] describe Access Control as being comprised of models on three different layers, **P**olicy, **E**nforcement and **I**mplementation (PEI), as illustrated in Figure 2.4. Policy models are used to formalize high level Access Control requirements, enforcement level models describe how to enforce these policies from a systems perspective, and the implementation level models show how to implement the components and protocols described by the enforcement model. In short we can say that P-models decide what requirement can be described, whereas the E- and I-models describe how to enforce the requirements.

There is a number of guiding principles for Access Control, first described by Saltzer at al. [62], the most notable ones being:

1. **Least privilege**, requires that a subject should only have the least privileges possible to perform its tasks.

2. **Separation of duties**, meaning that different subjects should have different tasks, e.g., an administrator should not also be an application user.

3. **Complete mediation** requires that any access to a resource must be monitored and verified.

The first two of these principles are related to the policy-layer, the last one is related to the enforcement layer.

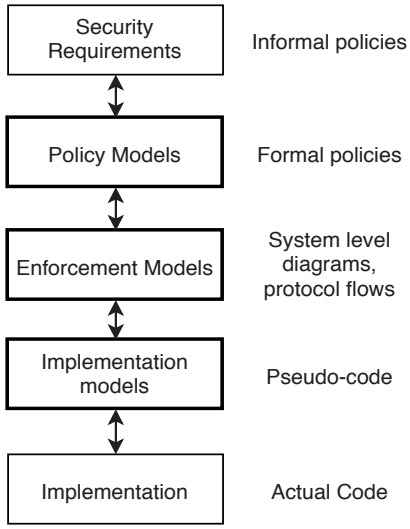


Figure 2.4: A PEI-model [63]

2.3.2 Policy Models

Access Control Policy models provide a formalized way to express a security policies. Which policies can be described are limited by the entities and primitives available in the model. In general, the complexity and flexibility of the system must be mirrored by an equal complexity and granularity available in the policy model in order to follow e.g., the principle of least privilege.

Historically, Mandatory Access Control (MAC) and Discretionary Access Control (DAC) have been the two main policy models within Access Control. MAC is based on security classifications on resources, combined with security clearances for subjects, e.g., top-secret content only readable for subjects with the highest security clearance. In DAC on the other hand, the privileges are defined as a relation between the resource and subject, often with the subject allowed to transfer its privileges.

Role-Based Access Control (RBAC) is building on principles from both DAC and MAC, where subjects have one or several roles that may be hierarchically ordered. Privileges are derived from the roles rather than from the subject. Currently RBAC is the most widely used model for Access Control [15], being used e.g., in the Windows Active Directory. Criticism on RBAC has however been raised [40, 84], noting that it does not allow for several seemingly simple use cases without an explosion in the amount of roles and groups required [28, 11]. It has also been noted that the concept of roles and groups is not a good

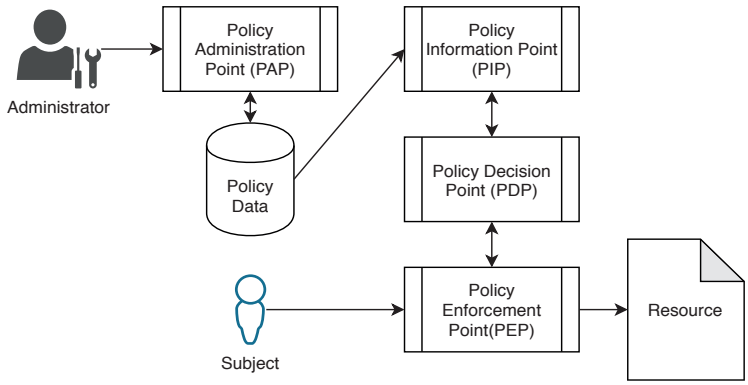


Figure 2.5: Authorization Enforcement Architecture

fit for use-cases including machine to machine interactions, as roles are not a natural concept for technical entities in the same way as for humans.

Yuan et al. [84] provide an early example of Attribute Based Access Control (ABAC) in 2005, in the article "Attribute Based Access Control (ABAC) for web services", as an alternative solution to the concerns raised against RBAC. In ABAC the policy rules are described using logical expressions based on attributes for subject, object and environment respectively. This allows for expression of extremely fine-grained policies. However, the management effort and transparency of expressed policies are challenging. There are currently two main enforcement models for ABAC, one based on the OASIS standard XCAML [83], and the other based on the NIST standard NGAC [13].

2.3.3 Enforcement Architecture

For an Access Control mechanism to be effective, all actions on resources must be mediated by entities able to decide and enforce the rules expressed in the policy model, as expressed by the principle of complete mediation. An example of such an enforcement architecture, often used in the context of ABAC [36, 13, 83], is depicted in Figure 2.5.

According to this architecture, all access to a resource must be mediated by a Policy Enforcement Point (PEP), responsible for protecting the resource and only allow legitimate requests. The PEP queries a Policy Decision Point (PDP), which is responsible for privilege inference, based on the subject and object identities, available policies, and possibly other policy-related information (such as attributes for the subject, object or environment). The PDP read policy through the Policy Information Point (PIP). Policy data is administered

through a Policy Administration Point (PAP). Placement and implementation of these entities are crucial for the functionality the Access Control mechanism as a whole can provide.

2.3.4 Access Control in IACS and the IIoT

In traditional IACS, the focus of Access Control has been mainly related to authorizing human users on technical assets. In older systems, physical access to an asset HMI combined with e.g., a PIN-code has been a sufficient level of control, as that was the only method for interacting with the device. In modern networked supervisory and control systems, coherent and transparent Access Control policies and enforcement frameworks are still largely inadequate [22]. Typically no special authorization rules are set up or even supported by the protocols for inter-device communication in the control network, instead communication is limited through configuration of the devices [32]. The rationale behind this set up is that it is easy to administer, since the users of the system is a quite limited number of employees, the network perimeters for the control networks are seen as well protected, and the devices within the network are limited to interact based on hard-wired interconnections.

In OPC UA there are available solutions related to device and service interactions, e.g., for providing proof of origin (using certificates issued through a public key infrastructure (PKI)) and including guidelines on e.g., auditing. There is however very limited technical support and guidance for Access Control [78], the explicit strategy stating that Access Control is an issue for the application developer to solve [25].

Considering the OT/IT convergence and the characteristics of the evolving systems within the Industry 4.0 paradigm, this rationale is worth reconsidering. The set of stakeholders and users within the system are high, including users outside the organization, the interconnections between components and services are not predefined, and networks are far from air-gapped, with some components in the system using ubiquitous connection protocols including wireless, etc., increasing the risk for a device getting compromised. The system complexity and its heterogeneous nature will make a compromised device more difficult to detect. Combined with a coarse-grained, or indeed missing, Access Control mechanism for inter-devices communication the risk associated with a compromised device launching a privilege escalation attack is high. Therefore, we come to the conclusion that the practice of including a strict Access Control between devices (and services) in modern IACS are important. The increasing amount of cybersecurity threats to these systems

makes the likelihood of a security breach higher, and the systems are also more dynamic and complex - e.g., it may not be predefined which parts of the system will need to interact during system design, and therefore a higher degree of flexibility is also required from the privilege handling mechanisms.

As the use-cases for inter-device Access Control within IIoT-systems requires a high level of flexibility and granularity, and the amount of devices and services are expected to be high, policy formulation with the aim of fulfilling the principle of least privilege will be a challenge. With the dynamic nature of the systems, the management of policies and related data are consequently expected to be complex and costly. Both these issues are impediments to inclusion of state of the art fine grained Access Control within modern IACS and IIoT systems.

Chapter 3

Research Summary

This chapter discusses the research process and presents the main research goals guiding the research work. It also briefly describes the approach and methods used to work towards the identified goals.

3.1 Research Process

The research process in this thesis can be seen as a set of iterative steps inspired by the Design Research Methodology (DRM) [7], see Figure 3.1. According to the DRM processes, the initial stage of the research is to perform a *research clarification*, to reach an understanding of the subject, and establish meaningful goals for future research. Articles A and B, included in this thesis, are produced as a part of this stage. In the second stage, a *descriptive study* is performed, leading to a deeper understanding related to the formulated goals in the first stage. Article C can be seen as such a descriptive study result. The third stage of DRM is a *prescriptive study*, where an improvement related to the studied area is suggested, in this thesis presented in Article D. The final stage of the process is a *second descriptive study*, where the improvement suggested in stage three is evaluated. Publications related to evaluations are planned as future work, and not included in this thesis.

The DRM process can also be applied at each single work package, meaning that the content of each produced article contains ingredients from all the stages, with goal clarifications, a detailed background related to the subject, contributions, and related validations. To validate results included in this research, a number of methods has been used. For literature surveys,

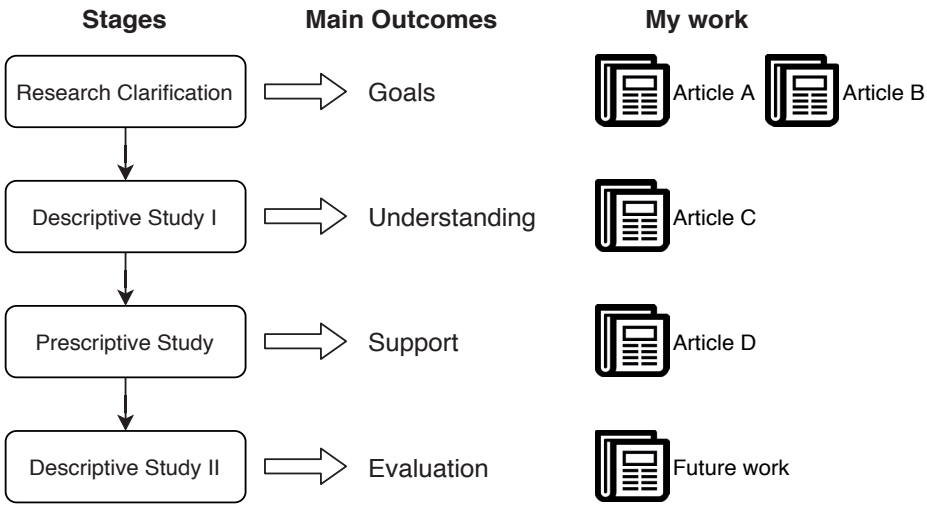


Figure 3.1: The Design Research Methodology process, related to included articles.

the Structured Literature Review (SLR) [30] method developed by Kitchenham is used as an inspiration, with a limitiation that not all aspects of SLR are considered. For case studies, we have used the checklist developed by Runesson et al. [58]. Verification based on proof by induction has been used to validate algorithm correctness. All of the resulting publications have also been discussed and reviewed by industrial experts.

3.2 Research Goals

Industrial control systems form the integration point between the physical and cyber world, controlling and supervising our most important and critical infrastructures, such as power utilities, clean water plants and nuclear plants, as well as the manufacturing industries at the basis of our economy. These systems are currently undergoing a transformation driven by the Industry 4.0 revolution. As a consequence, the cybersecurity threat landscape for industrial control systems is evolving as well. Being aware that a potentially malicious intruder exist and trying to minimize the harm a malicious intruder can cause, are two important mechanisms for addressing Cybersecurity challenges in industrial control systems. Development, study and dissemination of methods providing solutions for cybersecurity challenges are therefore of great importance for increasing the trustworthiness of the industrial control systems of today and tomorrow.

To that avail, we have formulated the following research goals, with an overall aim to increase the resilience and reliability of industrial control systems in the context of Industry 4.0:

RG1 To identify the gaps in current state of the art and industrial practices for cybersecurity in industrial control systems with regards to the emerging IIoT.

RG2 To identify the cybersecurity-related requirements on an IIoT system.

RG3 To propose new methods to enable dynamic Access Control in flexible industrial systems, such as Modular Automation.

3.2.1 Research Goal 1

The initial intuition leading to this work was that there are great challenges at the intersection between cybersecurity, traditional industrial control systems and Industry 4.0. As a way to reach an understanding of these challenges, our aim is to identify the gaps in current state of the art and practice in this area. Standardization is an important aspect of industrial control system security, therefore assessment of gaps and improvement of international IACS cybersecurity standards in relation to Industry 4.0 is a key ingredient towards increasing the overall security of IIoT systems.

3.2.2 Research Goal 2

To understand how to handle defined gaps from RG1, as the next step we need to identify and analyze the requirements related to the areas where gaps and challenges have been found. RG2 is therefore a natural continuation of RG1, analyzing the requirements related to cybersecurity in an IIoT system.

3.2.3 Research Goal 3

One of the early findings in studying the emerging dynamic systems prescribed by Industry 4.0, is the increasing need for fine-grained Access Control in such systems. Considering an internal attacker, these systems are very vulnerable, especially for malicious inter-device interactions since a legitimate device typically is seen as trustworthy and no strict Access Control is enforced upon incoming requests from such a device. If enforcing Access Control policies strictly, i.e., adhering to the least-privilege principle, the management effort of formulating and upholding such policies quickly becomes a burden for the engineering staff. Therefore the third research goal evolved as a need to address the lack of methods to enable fine-grained Access Control in dynamic

manufacturing systems without extensive engineering overhead. In this context, dynamic Access Control implies that the formulated policies shall follow or be updated accordingly as the system components are re-combined when adapting to fluctuating manufacturing requirements.

Chapter 4

Contributions

In the following chapter we provide a brief overview of research results within this thesis. As the thesis is a collection of articles, each included article is described, and a summary of the contributions are presented in relation to the research goals.

As the motivation for the research have been formulated as research goals, the ambition of the research have been to work towards contributing to these goals. As the goals are expressed with a rather wide scope, complete fulfillment of goals have never been the ambition. The contributions of research in this thesis can be described as ”research products”, being the embodiment of the results. For each of the included articles one or more such product(s) are provided, enumerated in Table 4.1.

Article	ID	Contribution description
A	**C1**	An analysis of identified gaps in state of the art with regards to cybersecurity in IIoT systems.
	C2	An analysis of cybersecurity requirements on IIoT systems.
B	**C3**	An analysis on how the existing cybersecurity standard IEC 62443 can cater for gaps identified in C1.
C	**C4**	A list of of requirements on Access Control models in Smart Manufacturing systems.
D	**C5**	A recipe-based automatic Access Control policy generation algorithm for Modular Automation systems.
	C6	A formal proof of the correctness of the algorithm in C5.

Table 4.1: Article Contributions.

4.1 Included articles

The following section describes the articles included in this licentiate thesis, detailing their respective contribution towards the research goals. For all of these articles, I have been the main driver and writer, under supervision of the co-authors. I developed the ideas and methods, performed the studies, provided the analysis and wrote the articles, which were discussed and reviewed by co-authors.

4.1.1 Article A

Cybersecurity Challenges in Large IIoT Systems, Björn Leander, Aida Čaušević, Hans Hansson, In proceedings of IEEE Emerging Technologies and Factory Automation (ETFA) 2019, Special session on Cybersecurity in Industrial Control Systems

Summary: In this article we derive high-level cybersecurity requirements on IIoT Systems, using the STRIDE threat model method on an industrial use case scenario. The requirements are then described through a state-of-the art review, and perceived challenges in relation to the requirements are discussed. The article contributes with an enumeration of cybersecurity requirements with regards to IIoT systems (C1), highlighting some of the main challenges that Industry 4.0 has already introduced in this context (C2).

4.1.2 Article B

Applicability of the IEC 62443 standard in Industry 4.0 / IIoT, Björn Leander, Aida Čaušević, Hans Hansson, In proceedings of International Conference on Availability, Reliability and Security (ARES) 2019, Workshop on Industrial Security and IoT (WISI).

Summary: This article is an artifact case study, were the IEC 62443 standard is analyzed in the light of the emerging systems of Industry 4.0. We identify some aspects of the standard which could prove difficult to reach compliance with in the context of Industry 4.0. For example, handling of cross-zone communication and secure software updates are areas in need of additional guidance. The article contributes to C3 by an analysis of the IEC 62443 standard in relation to the challenges posed by the Industry 4.0 evolution, as discussed in Article A.

4.1.3 Article C

Access Control in Smart Manufacturing Systems, Björn Leander, Aida Čaušević, Hans Hansson, Tomas Lindström, In proceeding of the 14th European Conference on Software Architecture (ECSA) 2020, 2nd Workshop on Systems, Architectures, and Solutions for Industry 4.0 (SASI4).
Summary: In this article, we discuss the need for fine-grained Access Control within Smart Manufacturing systems. We derive a set of requirements on Access Control models within such systems, being the main contribution (C4), based on the analysis of a literature study. Furthermore, the Attribute Based Access Control (ABAC) model is evaluated against the requirements, and found to be a potential candidate for use in such systems. As an illustration, we provide examples of how to use ABAC to describe certain types of rules within a Smart Manufacturing use case.

4.1.4 Article D

A Recipe-based Algorithm for Access Control in Modular Automation Systems, Björn Leander, Aida Čaušević, Hans Hansson, MRTC Report, Mälardalen Real-Time Research Centre, Mälardalen University, 2020.
Summary: We study the interactions between devices in Modular Automation systems, in order to understand how to express Access Control policies within such systems. The work is inspired by the conclusions from Article C, describing expression and management of fine-grained policies as one of the big challenges for Access Control within flexible modular systems. Using workflows expressed as Sequential Function Charts (SFC), we define a formal requirement on Access Control policies that must be fulfilled when using the Next Generation Access Control (NGAC) standard, and present an algorithm generating policies fulfilling that requirement. The article contributes through the algorithm for generation of Access Control policies for recipe orchestration (C5), along with a formal proof of its correctness (C6).

4.1.5 Contribution Summary

To summarize, the research products enumerated and described above, contributes to the research goals as illustrated in Table. 4.2.

RG1, aiming at identifying gaps in cybersecurity practices within IACS, are contributed to by a state of the art analysis on a high level (C1) and an analysis

Goal \ Article	A		B	C	D	
	C1	C2	C3	C4	C5	C6
RG1	x		x			
RG2		x		x		
RG3				x	x	x

Table 4.2: A mapping between research contributions with respect to the identified research goals.

of the IEC 62443 standard (C3). RG2, aiming at understanding cybersecurity requirements on the emerging IIoT-systems, are contributed to by a high level analysis of requirements (C2) and a list of requirements looking specifically on Access Control models in Smart Manufacturing (C4). C5 and C6, being an automatic algorithm for generating Access Control policies in modular automation systems contribute to RG3, which aims at proposing new methods to enable dynamic Access Control in flexible industrial systems. C4 also contributes to RG3, by defining the requirements for Access Control in such systems.

Chapter 5

Related Work

Three main areas are covered in this thesis, that are: cybersecurity in the context of Industry 4.0, dynamic Access Control, and Smart and Modular Manufacturing Systems. In this chapter we describe relevant academic efforts within these areas.

5.1 Cybersecurity in Industry 4.0

In the area of cybersecurity in Industry 4.0, there is a huge body of research, all of which cannot be reiterated here. Therefore our aim is to discuss the most relevant ones for our research topic. Several works are discussing current and future challenges related to the IoT, e.g., Frustaci et al. [16], Chiang et al. [8], Sadeghi et al. [59] and Sajid et al. [60].

Chiang et al. [8] discuss several fundamental challenges in using traditional cloud technology within the emerging IoT, and provide arguments for using fog nodes to counteract some of these challenges, e.g., related to latency requirements, bandwidth constraints, intermittent connectivity, etc. The focus is IoT in broad terms, including both consumer and industrial applications. A number of security related challenges are discussed. Frustaci et al. [16], provide a thorough analysis of current state of the art for securing IoT devices and data, as well as an evaluation of identified critical security issues related to IoT.

Sadeghi et al. [59] discuss challenges in Industrial IoT systems with regards to security and privacy, arguing that current available security solutions for IoT must be enhanced with mechanisms that scale better with the large and

heterogeneous systems of IIoT. Sajid et al. [60] provides a review of security challenges and state of the art solutions for Cloud-assisted IoT solutions in SCADA, together with suggestion for future research. Challenges for both traditional SCADA systems and Cloud-connected SCADA are described.

Even though there are several similarities between our work and the ones described above regarding identified challenges, none of the studied works use threat modeling as a method for requirement inference as we have proposed. Threat modeling is one of the methods traditionally used to identify weak spots in IT systems, being of increasing use in IACS, mandated by the IEC 62443 [24] standard[1]. Therefore, it is a natural approach to use this method as a starting point for discussing challenges arising from identified threats.

Compliance to industrial cybersecurity standards are of great importance for IACS as well as other safety critical systems, due to possible HSE impact on a successful breach of security. Yet no scientific work has been found specifically assessing the IEC 62443 standard (or any other relevant cybersecurity standard) with regards to Industry 4.0. Our research makes an attempt at filling that gap.

5.2 Dynamic Access Control

As there are limited academic research specifically investigating dynamic Access Control within industrial control systems, we have additionally examined research related to Access Control for similar systems. All of the below described works have ingredients of interest and bear similarities with our work, none of them are however directly applicable for use in IACS.

Some of the existing work present variations of ABAC suitable in different domains. Lang et al. [36] suggest a Proximity Based Access Control (PBAC), well suited for e.g., intelligent transportation systems. It originates from the ABAC model, but uses the mathematical proximity between a subject and a resource as one of the deciding factors for granting privileges. Park and Sandhu [52] present the Usage CONtrol (UCON$_{ABC}$)-model, which can also be seen as an extension of the ABAC model, but includes obligations. In this approach one privilege request could alter attributes or conditions for future Access Controls. This mutability of attributes, or a variation thereof, could possibly be used to model the behavior of temporal workflows required by smart manufacturing. The models are therefore of interest, but there is no

[1]IEC 62443-4-1 System Requirement 2

guidance on how to formulate policies to handle such workflows, which we make an attempt at in our work.

In the field of Model-Driven Security (MDS), originating from Model Driven Architecture [31], there is a lot of previous research related to the design of secure systems, with regards to modeling, analysis as well as model transformation. Basin et al. [6], summarize a large portion of the existing work related to this topic. The focus of MDS is mainly on the design phase for including security specific models when realizing a system architecture, by e.g., defining modeling languages for Access Control rules [41]. Most of MDS research is, with regards to Access Control, focused on the RBAC-model. However there are some examples utilizing attribute based Access Control. Such example is provided by Alam et al. [2], describing an MDS approach for SOA, with XACML as policy expression language. As MDS is using system models as input data, it differs from our approach, using workflow models as input data.

Task-Based Authorization Control (TBAC) [72] is an Access Control model aimed at limiting privileges to a just-in-time and need-to-do basis, being similar objectives as what we try to reach for authorization within modular manufacturing systems. The idea is to have a set of trustees validating each privilege request, and granting privileges will be limited also by expected usage, e.g., number of allowed resource accesses. However, as far as we understand, TBAC never materialized in any expression language or reference implementation, making it an unfeasible choice for an industrial system. In our work we try to reach the same objectives, but using a standardized expression language for the policies.

5.3 Smart and Modular Manufacturing systems

Some previous academic research within the area of cybersecurity in smart manufacturing and similar systems are closely related to our work. Below we discuss and position that work in relation to ours.

Salonikias et al. [61] and Lopez et al. [42] discuss requirements on Access Control models in IIoT systems and cyber-physical systems from the policy level perspective. However, they do not consider the modular and dynamic features of smart manufacturing and modular automation, which is one of the major challenges targeted by our work.

Watson et al. [78], discusses the use of a number of different Access Control models in conjunction with OPC UA. The authors advocate the use of ABAC or a combination of ABAC and RBAC as a good match for protection against

privilege escalation for both inside and outside attackers within IACS. Ruland et al. [57] describe an XACML based Access Control system for smart energy grids, including attributes related to system state, allowing for some amount of dynamicity with regards to privilege deduction. The main usage of the system state is as a conditional for safety related functionality. Both these works touch upon our suggested solutions for Access Control policy formulations, but none of them supports use cases related to dynamic system composition, being one characteristic we aim to enable in our research.

Tuptuk et al. [73] and Waidner et al. [77] discuss several challenges related to cybersecurity and smart manufacturing systems, in general arguing that too little attention is paid to cybersecurity in current related academic research. Nonetheless, there are several works looking at specific problems and solutions in this area, e.g., Alcaraz [3] looking at secure IT-OT interconnections in Industry 4.0, and Preuveneers et al. [55] looking at identity management in smart manufacturing systems. Although all these works are of relevance, none look at Access Control policy formulation and management, which is our contribution in the area.

Seifert et al. [66] and Ladiges et al. [35] describe the concept and current state of Modular Automation, but do not specifically discuss emerging cybersecurity threats in relation to these systems. We expand the understanding of modular automation systems by describing some of the threats, related to privilege escalation attacks. Moreover, we discus Access Control policy requirements on machine to machine interactions being a mitigating measure, as well as suggest a method for generation of such policies.

Chapter 6

Conclusions

6.1 Summary of contributions

In this licentiate thesis we have discussed a number of challenges related to cybersecurity in systems evolving within the Industry 4.0 paradigm. Within the scope of the thesis we have provided a list of the high level requirements on IIoT systems, and discussed how well the leading industry standard for cybersecurity caters for these requirements. One of the main conclusions from the state of the art and standard study relates to the management and formulation of Access Control policies. Management of security properties becomes increasingly cumbersome to deal with it in these systems, especially for use cases were the system components are continuously re-organized, such as in the case for Smart Manufacturing and Modular Automation. A number of requirements for Access Control models within smart manufacturing systems are therefore provided and one of the novel models, ABAC, is evaluated in relation to the requirements. Furthermore, an algorithm for generating Access Control policies within Modular Automation systems is developed and formally verified.

6.2 Future directions

As a continuation of the work related to automatic policy generation, there is a need to implement the algorithm and possibly include it in a real systems. Working across all the PEI-layers within the flexible systems of Industry 4.0 is a potential way forward that might provide improvements on Access Control as well as on the enforcement architecture and component implementation.

This extension is necessary in order to reach industrially feasible solutions, as the currently achieved results cannot be evaluated properly without inclusion in an enforcement architecture.

There are clearly other paths forward that can be examined while staying at the policy-model layer, e.g., development of policy generative algorithms for other applications, or other Access Control models.

Within the vast research area of cybersecurity issues related to Industry 4.0, an immense amount of work remains. In one of the included articles we discuss the need for automated secure software and firmware updates for devices and services within industrial control systems, a task traditionally requiring substantial effort from engineering personnel. This effort is foreseen to mount with increasing numbers and heterogeneity of devices and services within IACS, as a result of the Industry 4.0 paradigm. Therefore, there is substantial work needed, both in the fields of standardization and method development in the area of secure software updates.

Bibliography

[1] Control Systems Cyber Security:Defense in Depth Strategies. *Idaho National Laboratory, USA*, (May):8, 2006.

[2] M. Alam, R. Breu, and M. Hafner. Model-driven security engineering for trust management in SECTET. *Journal of Software*, 2(1):47–59, 2007.

[3] C. Alcaraz. Secure Interconnection of IT-OT Networks in Industry 4.0. pages 201–217, 2019.

[4] R. Baheti and H. Gill. Cyber-pysical Systems. *The impact of control technology*, 12(1):161–166, 2011.

[5] S. B. Baker, W. Xiang, and I. Atkinson. Internet of things for smart healthcare: Technologies, challenges, and opportunities. *IEEE Access*, 5:26521–26544, 2017.

[6] D. Basin, M. Clavel, and M. Egea. A decade of model-driven security. In *Proceedings of the 16th ACM Symposium on Access Control Models and Technologies*, SACMAT '11, page 1–10, New York, NY, USA, 2011. Association for Computing Machinery.

[7] L. T. Blessing and A. Chakrabarti. *DRM: A design reseach methodology*. Springer, 2009.

[8] M. Chiang and T. Zhang. Fog and IoT: An Overview of Research Opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.

[9] J. Davis, T. Edgar, J. Porter, J. Bernaden, and M. Sarli. Smart manufacturing , manufacturing intelligence and demand-dynamic performance. *Computers and Chemical Engineering*, 47:145–156, 2012.

[10] Z. Drias, A. Serhrouchni, and O. Vogel. Analysis of cyber security for industrial control systems. In *2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*, pages 1–8, 2015.

[11] A. Elliott and S. Knight. Role Explosion: Acknowledging the Problem. *Software Engineering Research and Practice, WORLDCOMP*, (October 1992):349–355, 2010.

[12] J. P. Farwell and R. Rohozinski. Stuxnet and the future of cyber war. *Survival*, 53(1):23–40, 2011.

[13] D. Ferraiolo, R. Chandramouli, R. Kuhn, and V. Hu. Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC). pages 13–24, 2016.

[14] FIRST. Common Vulnerability Scoring System. `https://www.first.org/cvss/`, 2019. [Online; accessed 29-may-2019].

[15] V. N. L. Franqueira and V. F. I. Consulting. Kent Academic Repository Role-Based Access Control in Retrospect. 45:81–88, 2012.

[16] M. Frustaci, P. Pace, G. Aloi, and G. Fortino. Evaluating critical security issues of the IoT world: Present and future challenges. *IEEE Internet of Things Journal*, 5:2483–2495, 2018.

[17] A. Greenberg. *Sandworm: A New Era of Cyberwar and the Hunt for the Kremlin's Most Dangerous Hackers*. Random House Books for Young Readers, 2019.

[18] R. Henßen and M. Schleipen. Interoperability between OPC UA and AutomationML. *Procedia CIRP*, 25(C):297–304, 2014.

[19] M. Hermann, T. Pentek, and B. Otto. Design principles for industrie 4.0 scenarios. In *Proceedings of the Hawaii International Conference on System Sciences*, volume 2016-March, pages 3928–3937. IEEE, 2016.

[20] S. H. Houmb, V. N. Franqueira, and E. A. Engum. Quantifying security risk level from CVSS estimates of frequency and impact. *Journal of Systems and Software*, 83(9):1622–1634, 2010.

[21] M. Howard and S. Lippner. *Security Development Life-Cycle*. Microsoft Press, 2006.

[22] J. H. Huh, R. B. Bobba, T. Markham, D. M. Nicol, J. Hull, A. Chernoguzov, H. Khurana, K. Staggs, and J. Huang. Next-Generation Access Control for Distributed Control Systems. *IEEE Internet Computing*, 20(5):28–37, 2016.

[23] IEC. Smart Manufacturing - Reference Architecture Module Industry 4.0 (RAMI4.0). Technical report, Internation Electrotechnical Commission, 2016.

[24] IEC 62443 security for industrial automation and control systems. Standard, Internation Electrotechnical Commission, Geneva, CH, 2009-2018.

[25] IEC 62541 OPC unified architecture. Standard, Internation Electrotechnical Commission, Geneva, CH.

[26] IIC. The Industrial Internet of Things Volume G4 : Security Framework. Technical report, Industrial Internet Consortium, 2016.

[27] IIC. The Industrial Internet of Things Volume G1: Reference Architecture. Technical Report November, Industrial Internet Consortium, 2017.

[28] X. Jin, R. Sandhu, and R. Krishnan. RABAC: Role-centric attribute-based access control. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7531 LNCS:84–96, 2012.

[29] R. Kissel. *Glossary of key information security terms, Revision 2.* U.S. Dept. of Commerce, National Institute of Standards and Technology, 2013.

[30] B. A. Kitchenham. Procedures for Undertaking Systematic Reviews. Technical report, Keele University, 2004.

[31] A. G. Kleppe, J. Warmer, J. B. Warmer, and W. Bast. *MDA explained: the model driven architecture: practice and promise.* Addison-Wesley Professional, 2003.

[32] E. D. Knapp and J. T. Langill. *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems.* Syngress, 2014.

[33] W. Knowles, D. Prince, D. Hutchison, J. F. P. Disso, and K. Jones. A survey of cyber security management in industrial control systems. *International Journal of Critical Infrastructure Protection*, 9:52 – 80, 2015.

[34] A. Kusiak. Service manufacturing: Basic concepts and technologies. *Journal of Manufacturing Systems*, 52:198–204, 2019.

[35] J. Ladiges, A. Fay, T. Holm, U. Hempen, L. Urbas, M. Obst, and T. Albers. Integration of modular process units into process control systems. *IEEE Transactions on Industry Applications*, 54(2):1870–1880, March 2018.

[36] U. Lang and R. Schreiner. Proximity-based access control (PBAC) using model-driven security. In H. Reimer, N. Pohlmann, and W. Schneider, editors, *ISSE 2015*, pages 157–170, Wiesbaden, 2015. Springer Fachmedien Wiesbaden.

[37] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann. Industry 4.0. *Business & information systems engineering*, 6(4):239–242, 2014.

[38] R. Leszczyna, E. Egozcue, L. Tarrafeta, V. F. Villar, R. Estremera, and J. Alonso. Protecting industrial control systems-recommendations for eu-

rope and member states. *tech. rep., Technical report, European Union Agency for Network and Information Security (ENISA)*, 2011.

[39] J. Q. Li, F. R. Yu, G. Deng, C. Luo, Z. Ming, and Q. Yan. Industrial Internet: A Survey on the Enabling Technologies, Applications, and Challenges. *IEEE Communications Surveys and Tutorials*, 19(3):1504–1526, 2017.

[40] N. Li, J. W. Byun, and E. Bertino. A critique of the ANSI standard on role-based access control. *IEEE Security and Privacy*, 5(6):41–49, 2007.

[41] T. Lodderstedt, D. Basin, and J. Doser. SecureUML: A UML-based modeling Language for model-driven security. In *International conference on model engineering, concepts and tools*, 2002.

[42] J. Lopez and J. E. Rubio. Access control for cyber-physical systems interconnected to the cloud. *Computer Networks*, 2018.

[43] Y. Lu. Industry 4.0: A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration*, 6:1 – 10, 2017.

[44] Y. Lu and F. Ju. Smart Manufacturing Systems based on Cyber-physical Manufacturing Services (CPMS). *IFAC-PapersOnLine*, 50(1):15883–15889, 2017.

[45] W. Madsen. *Trust in Cyberspace*. National Academies Press, 1999.

[46] Microsoft. The STRIDE Threat Model. `https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)`, 2005. [Online; accessed 5-march-2019].

[47] S. Mittal, M. A. Khan, and T. Wuest. Smart manufacturing: Characteristics and technologies. In R. Harik, L. Rivest, A. Bernard, B. Eynard, and A. Bouras, editors, *Product Lifecycle Management for Digital Transformation of Industries*, pages 539–548, Cham, 2016. Springer International Publishing.

[48] S. Mohurle and M. Patil. A brief study of Wannacry Threat: Ransomware Attack 2017. *International Journal of Advanced Research in Computer Science (IJARCS)*, 8(5):1938–1940, 2017.

[49] S. Mumtaz et al. Massive Internet of Things for Industrial Applications: Addressing Wireless IIoT Connectivity Challenges and Ecosystem Fragmentation. *IEEE Ind. Elec. Magazine*, 11(1), 2017.

[50] S. Myagmar, A. J. Lee, and W. Yurcik. Threat modeling as a basis for security requirements. In *Symposium on requirements engineering for information security (SREIS)*, volume 2005, pages 1–8. Citeseer, 2005.

[51] P. Neirotti, A. De Marco, A. C. Cagliano, G. Mangano, and F. Scorrano. Current trends in smart city initiatives: Some stylised facts. *Cities*, 38:25 – 36, 2014.

[52] J. Park and R. Sandhu. The UCON$_{ABC}$ usage control model. *ACM Transactions on Information and System Security*, 7(1):128–174, 2004.

[53] R. S. H. Piggin. Emerging good practice for cyber security of industrial control systems and SCADA. In *7th IET International Conference on System Safety, incorporating the Cyber Security Conference 2012*, pages 1–6, 2012.

[54] E. G. Popkova, Y. V. Ragulina, and A. V. Bogoviz. *Fundamental Differences of Transition to Industry 4.0 from Previous Industrial Revolutions*, pages 21–29. Springer International Publishing, Cham, 2019.

[55] D. Preuveneers, W. Joosen, and E. Ilie-Zudor. Identity management for cyber-physical production workflows and individualized manufacturing in Industry 4.0. *Proceedings of the ACM Symposium on Applied Computing*, Part F1280:1452–1455, 2017.

[56] M. Rocchetto and N. O. Tippenhauer. On attacker models and profiles for cyber-physical systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9879 LNCS:427–449, 2016.

[57] C. Ruland and J. Sassmannshausen. Access Control in Safety Critical Environments. In *Proceedings - 12th International Conference on Reliability, Maintainability, and Safety, ICRMS 2018*, pages 223–229. IEEE, 2018.

[58] P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131, Dec 2008.

[59] A.-R. Sadeghi, C. Wachsmann, and M. Waidner. Security and privacy challenges in industrial internet of things. *Proceedings of the 52nd Annual Design Automation Conference on - DAC '15*, pages 1–6, 2015.

[60] A. Sajid, H. Abbas, and K. Saleem. Cloud-Assisted IoT-Based SCADA Systems Security: A Review of the State of the Art and Future Challenges. *IEEE Access*, 4:1375–1384, 2016.

[61] S. Salonikias, A. Gouglidis, I. Mavridis, and D. Gritzalis. Access control in the industrial internet of things. In C. Alcaraz, editor, *Security and Privacy Trends in the Industrial Internet of Things*. Springer International Publishing, 2019.

[62] J. Saltzer and M. Schroeder. The Protection of Information in Computer Systems. In *proceedings of the IEEE*, volume 63, pages 1278–1308, September 1975.

[63] R. Sandhu, K. Ranganathan, and X. Zhang. Secure information sharing enabled by trusted computing and PEI models. *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS '06*, 2006:2–12, 2006.

[64] R. S. Sandhu and P. Samarati. Access control: Principles and Practice. *IEEE Communications Magazine*, 32(September):40–48, 1994.

[65] S. Schriegel, T. Kobzan, and J. Jasperneite. Investigation on a distributed SDN control plane architecture for heterogeneous time sensitive networks. *IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS*, 2018-June:1–10, 2018.

[66] T. Seifert, S. Sievers, C. Bramsiepe, and G. Schembecker. Small scale, modular and continuous: A new approach in plant design. *Chemical Engineering and Processing: Process Intensification*, 52:140–150, 2012.

[67] J. Slay and M. Miller. Lessons learned from the Maroochy water breach. In *International conference on critical infrastructure protection*, pages 73–82. Springer, 2007.

[68] J. Slowik. Evolution of ICS Attacks and the Prospects for Future Disruptive Events. Technical report, 2017.

[69] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn. Guide to Industrial Control Systems (ICS) Security NIST Special Publication 800-82 Revision 2. *NIST Special Publication 800-82 rev 2*, pages 1–157, 2015.

[70] M. Taddeo. Deterrence by Norms to Stop Interstate Cyber Attacks. *Minds and Machines*, 27(3):387–392, 2017.

[71] K.-d. Thoben, S. Wiesner, and T. Wuest. "Industrie 4.0" and Smart Manufacturing – A Review of Research Issues and Application Examples. *International Journal of Automation Technology*, (January), 2017.

[72] R. K. Thomas and R. S. Sandhu. Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented authorization management. In *Database Security XI*, pages 166–181. Springer, 1998.

[73] N. Tuptuk and S. Hailes. Security of smart manufacturing systems. *Journal of Manufacturing Systems*, 47(April):93–106, 2018.

[74] S. Vandermerwe and J. Rada. Servitization of business: Adding value by adding services. *European Management Journal*, 6(4):314 – 324, 1988.

[75] B. Vogel-Heuser and D. Hess. Guest Editorial Industry 4.0–Prerequisites and Visions. *IEEE Transactions on Automation Science and Engineering*, 13(2):411–413, 2016.

[76] W. Voorsluys, J. Broberg, R. Buyya, et al. Introduction to cloud computing. *Cloud computing: Principles and paradigms*, pages 1–44, 2011.

[77] M. Waidner and M. Kasper. Security in industrie 4.0 - Challenges and solutions for the fourth industrial revolution. *Proceedings of the 2016 Design, Automation and Test in Europe Conference and Exhibition, DATE 2016*, pages 1303–1308, 2016.

[78] V. Watson, J. Sassmannshausen, and K. Waedt. Secure Granular Interoperability with OPC UA. In C. Draude, M. Lange, and B. Sick, editors, *INFORMATIK 2019: 50 Jahre Gesellschaft für Informatik – Informatik für Gesellschaft (Workshop-Beiträge)*, pages 309–320, Bonn, 2019. Gesellschaft für Informatik e.V.

[79] M. H. Weik. *Computer Science and Communications Dictionary: Data integrity*, pages 350–350. Springer US, Boston, MA, 2001.

[80] M. Weyrich and C. Ebert. Reference Architectures for the Internet of Things. *IEEE Software*, 33:112–116, 2016.

[81] M. Whitman and H. Mattord. *Principles of Information Security*. Cengage Learning, 4th edition, 2012.

[82] T. J. Williams. The purdue enterprise reference architecture. *Computers in Industry*, 24(2):141 – 158, 1994.

[83] eXtensible Access Control Markup Language (XACML) version 3.0 plus errata 01. Standard, OASIS, 2017.

[84] E. Yuan and J. Tong. Attributed Based Access Control (ABAC) for web services. In *Proceedings - 2005 IEEE International Conference on Web Services, ICWS 2005*, volume 2005, pages 561–569, 2005.

Part II

Included Articles

Article A

Chapter 7

Article A: Cybersecurity Challenges in Large IIoT Systems

Björn Leander, Aida Čaušević, Hans Hansson
In proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, September 2019

Abstract

To achieve efficient and flexible production at affordable prices, industrial automation is pushed towards a digital transformation. Such a transformation assumes an enhancement of current Industrial Automated Control Systems with a large amount of IoT-devices, forming an Industrial Internet of Things (IIoT). The aim is to enable a shift from automatic towards autonomous control in such systems. This paper discusses some of the main challenges IIoT systems are facing with respect to cybersecurity. We discuss our findings in an example of a flow-control loop, where we apply a simple threat model based on the STRIDE method to deduce cybersecurity requirements in an IIoT context. Moreover, the identified requirements are assessed in the light of current state of the art solutions, and a number of challenges are discussed with respect to a large-scale IIoT system, together with some suggestions for future work.

7.1 Introduction

The manufacturing industry is going through a rapid evolution driven by the Internet technology applied in the industrial context. The paradigm shift is known as *Industry 4.0* in Europe and *Industrial Internet* in the USA. A common belief is that an emerging Industrial Internet of Things (IIoT) will provide optimization, cost-savings, and new business opportunities in several domains. According to the Industrial Internet Consortium (IIC) [18], an IIoT system will enable significant advances in optimizing decision-making, operations and collaborations among a large number of increasingly autonomous control systems. Big-data analysis using data from smart production equipment and smart products might for example provide intelligence for decision making. According to the IEC [20], a fundamental purpose of Industry 4.0 is to enable cooperation and collaboration between devices.

As described by Madsen [24], the trustworthiness of an information system is the degree of confidence that it performs as expected with respect to key characteristics during unexpected scenarios, such as: disruptions from the environment, human errors, system faults, and attacks from adversaries. An IIoT system will as well be judged based on its trustworthiness. The correct implementation of cybersecurity in an IIoT system will be one of the driving factors for its success, increasing its trustworthiness in several aspects such as: quality and integrity of information, asset availability, etc. However, many of the devices in an IIoT system will be resource constrained with regards to computational power, network bandwidth, etc., while there at the same time may be real-time requirements on signal handling. This combination of constraints and requirements yields unique challenges related to cybersecurity, as the traditional cryptographic methods add significant load both on network and CPU utilization.

Hermann et al. [17] describe the central design principles for Industry 4.0 as being: *1)* interconnection, *2)* technical assistance, *3)* decentralized decisions and *4)* information transparency. In this paper we mainly focus on interconnection, since reliable communication between devices in a heterogeneous environment is a fundamental requirement for enabling the remaining design principles.

The main contributions of our work are twofold: *1)* to uncover a number of cybersecurity related challenges in large-scale IIoT systems for which the current state of the art solutions need further improvements to be applicable, and *2)* presenting possible directions for future solutions for some of the more important of these challenges. We do this by applying the industry approved

Microsoft STRIDE [25] threat modelling method on a number of typical scenarios in a simple example. From the resulting threat model, information regarding cybersecurity threats related to an IIoT system are discussed.

The paper is organised as follows. Section 7.2 introduces necessary background and concepts used in this paper. In Section 7.3 a working example is introduced Section 7.4 expands the view to an IIoT system, including a threat model for the example based on STRIDE model, along with state of the art solutions for common countermeasures. In Section 7.5 we discuss challenges for a large-scale IIoT system from a cybersecurity perspective, while related works are described in Section 7.6. We present concluding remarks and outline directions for future work in Section 7.7.

7.2 Background

An IIoT system connects and integrates industrial control systems with enterprise systems, business processes and analytics. Boyes et al. [4] provide a more exhaustive definition of an IIoT system, based on a survey of existing definitions. This definition emphasize IIoT as a means for optimising overall production value. There exist several reference architectures related to IIoT, the most notable ones are: *Reference Architecture Module for Industry 4.0* (RAMI4.0) [20] suggested by IEC/PAS, and *Industrial Internet of Things Infrastructure* [19] suggested by the IIC.

For large scale IIoT applications, the complexity of the information infrastructure depends on:
1) ***System Size*** - In a factory or process industry there will be potentially many thousands or even millions of IIoT devices.
2) ***Composite devices*** - Complex devices will be composed of a number simpler devices, e.g., a smart mine hoist will consist of smart motors, transmission systems, brakes, sensors, etc.
3) ***Thing-to-Cloud Continuum*** - Different services related to specific devices or specific functions will exist anywhere from the device through edge nodes concentrating data to cloud nodes that collect and analyse data. For each device there could be any number of edge-, and cloud-nodes hosting related services, which will require communication and trust across organization boundaries in many applications.
4) ***Heterogeneous technologies*** - Many different manufacturers and industries using different technologies will implement and use these devices. At the same time, the devices are expected to be able to communicate with other devices and services along the thing-to-cloud continuum when needed. Inter-

operability between devices and services will be a paramount.
*5) **Multiple stakeholders*** - Different stakeholders will have interest in the devices, including device owner, device manufacturer, maintenance responsible, etc.
Therefore a large-scale IIoT system has advanced requirements on the information infrastructure. It will become an important task to address different levels of integration required in an IIoT infrastructure, as described in [27]:

1. Cross-technology integration of smart devices from different suppliers;
2. Cross-organization integration of information and services from different enterprises;
3. Cross-domain integration of business ecosystems from different industries.

Cybersecurity is the protection of a computer system from unauthorized actors' possibility to: steal or alter information in the system, disrupt or alter behaviour of a function or perform an unauthorized action [22]. Cybersecurity is seen as a cross-cutting concern of an IIoT system [18], as a system is not more secure than its weakest link, and any potential attack surface must be considered.

The STRIDE (**S**poofing, **T**ampering, **R**epudiation, **I**nformation disclosure, **D**enial of Service and **E**levation of privilege) threat model is a method for classifying threats in an information system introduced by Microsoft [25]. It includes defining security zones in a data-flow diagram for the system, checking any security-zone interactions and then enumerating any threat per class for that interaction. For each threat, countermeasures are suggested and assessed. The use of STRIDE for threat modeling in IIoT has already been discussed in the literature [18], referring to an extension of STRIDE for the Azure IoT reference architecture [26], described by Shahan et. al [30]. Other possible methods for threat modeling could be considered, such as CVSS [11], PASTA [32], etc. As STRIDE is commonly used in industry it was selected for this work.

In this paper the focus is on a class of assets that in RAMI4.0 is defined as an *entity*, being an uniquely identifiable asset that has a digital world representation. *Device* is in this context used interchangeable with entity. Focus is on information, however the devices may be used for sensing or actuating in the physical world. Such devices in combination with their software are realized as Cyber Physical Systems (CPS), which share a number of characteristics differentiating them from traditional IT-Systems. The main difference being

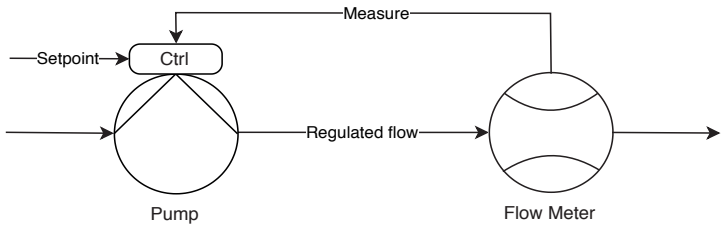


Figure 7.1: Flow-control loop

that actions for a CPS in the information-world can have direct real-world implications [2].

7.3 A working example

In this section we introduce a flow-control loop process as an example used to illustrate and derive challenges in the following work. It is chosen as being one of the simplest realistic control loops, common in industrial applications. The loop consists of a pump with built-in control logic that is regulating the flow through a pipe. The feedback is provided from a flow meter mounted in the pipe, see Fig. 7.1. This view of the process is only one example of a high-order integrated view of the control logic for the pump. Several other aspects exist for the pump in a practical industrial application, e.g., CAD drawing, location, current I/O values, status, graphics, maintenance log, I/O-value history, etc. In an Industrial Automated Control System (IACS), these aspects are usually accessible in some way, but not always in the same view or related to the same identity.

The presented example can be seen as a CPS, and in section 7.4, we put the example into the context of a large IIoT system. In a CPS, cybersecurity attacks on the system might have physical-world implications, e.g., loss of control of the pump could harm the process and potentially pose threats to humans working in vicinity to the process, or to the environment, depending on the function of the system and additional safety measures supplied in the IACS.

Here we focus on three scenarios related to the flow-control loop from the perspective of a traditional Industrial Automation and Control System (IACS) (i.e., a homogeneous environment where different actors communicate using the same protocols, have a common identification nomenclature, and live within the same network). The scenarios are chosen as being typical events

in an industrial application.

7.3.1 Scenario 1 - Displaying a trend curve

An engineer wants to access current I/O-values from the pump and flow-meter in order to draw a trend-curve diagram to be displayed in a control room. The actions needed are: *1)* identify the pump and flow meter, *2)* check that there exist a service able to deliver relevant data for the respective device, *3)* use the service(s) to read the data, and *4)* display the trend-curve to the operator.

7.3.2 Scenario 2 - Replacing the pump device

In this scenario the pump in the flow control loop needs to be replaced due to some malfunction. To execute the scenario the required life-cycle actions of the old and new devices must be satisfied, including: *1)* a new pump must be acquired, *2)* the logical replacement in the IT-system, *3)* the physical replacement is executed by a technician on site, *4)* configuration of the new device must be performed so that it delivers the same functionality as the old pump. If there is substantial difference in functionality between the old and new devices, some services may need to be added or modified, e.g., the control logic could be implemented in a PLC if missing in the new pump-device.

7.3.3 Scenario 3 - Replace software in pump device

The pump manufacturer has discovered a fault or weakness in the current software version running on the pump device, requiring a patch being applied to resolve the issue. The scenario is executed in the following steps: *1)* the manufacturer of the pump notifies the plant organisation about a new patched software version for the pump-device, *2)* the patch is distributed to a maintenance technician, *3)* within a time-slot for planned maintenance a technician updates the pump device software.

7.4 A Threat Model from an IIoT perspective

Let us assume that the described example is a part of an IIoT system. Any aspect of the pump described in Section 7.3 could be represented by a separate service in this context. For example, a CAD-drawing related to the pump-device could be stored as a pdf-file directly in the device, accessible from a Product Life-cycle Management (PLM) system in the process owner's IT-network, or available at the pump manufacturer web-site. In this way each

device may be related to any number of services with endpoints distributed through the device-to-cloud continuum.

Scenarios 1 and 2, described above, will be performed in very much the same way in the IIoT perspective, only the environment will differ. For example, in Scenario 1, different services for I/O-data and for the trend curve might have endpoints in different security zones, communicate with different protocols, etc.

Scenario 3 on the other hand might differ substantially when performed in an IIoT system. The manufacturer might have direct access to some services related to the pump, e.g., a service containing information on current software version. The pump-device could have direct access to a service publishing new software revisions. Assuming previously described, the scenario will be executed as follows: *1)* the manufacturer publishes a new revision of the software containing the patch, *2)* then triggers the pump to perform an update, alternatively the pump-device could cyclically check for availability of updates, and finally *3)* the pump will download and perform the update automatically at a convenient time-slot.

Assuming an IIoT system setup, every device and service should be treated as being placed in separate security zones. In threat modeling every interaction crossing a security-zone boundary must be analyzed. A simple threat model for scenarios 1-3 using the STRIDE classification method is presented in Table 7.1. From this model, a number of additional requirements for devices and services that are part of an IIoT system can be deduced.

Note that the threat model is abstracted, a number of additional technical details should be accounted for when analyzing these scenarios in a system with specific protocols, operating systems, etc. Some aspects have been intentionally left out, e.g., physical security, threats from social engineering, etc., as they are not of interest for this work.

The additional security requirements, as listed in the threat model, can be sorted out based on the level of responsibility needed for their implementation:
1) ***Service:*** integrity and encryption of Data at Rest (DAR), hardening, resource limitation of unauthorized inbound connections, parameter bound checks, auditing.
2) ***Device:*** integrity and encryption of DAR, secure boot, function for purge of sensitive data, tamper free storage, anti-malware software, service sandboxing.

Classification	Scenario	Threat	Counter measure
Spoofing	1,3	Service-endpoint spoofed - impersonation attack.	Identification and authentication of service end-points.
	2	New Device is counterfeit.	Integrity of type-identification, policy on verification of authenticity of purchased products.
	1,3	Replay attack intended to trick a service to e.g., leak information.	Using e.g., session tokens to invalidate old messages.
Tampering	1,3	DIM tampered.	Integrity of DIM.
	1	DAR tampered	Integrity of DAR, tamper-free storage.
	2	Software of new device tampered.	Malware detection.
	3	Patch tampered during transfer.	Integrity check of patch before being applied, malware detection.
Repudiation	1,3	Device/Service claiming not received data/patch.	Audit log for accessing data.
	1,3	Device/Service claiming not sending data/patch.	Audit log for sending data.
	1,3	Actor claiming not attempting to access restricted information.	Audit log for failed access attempts.
Information Disclosure	1	Information intercepted and relayed to unintended receiver.	Encryption of DIM.
	2	Decommissioned device contains retrievable sensitive information.	Encryption of DAR, purge of disk/non-volatile memory, tamper free storage.
	2	Key material on decommissioned device could be used to decipher recorded network traffic.	Purge of cryptographic data, tamper free storage.
	2	Decommissioned device could be reconnected as a “rogue device” to intercept information.	Policy on revocation of decommissioned privileges.
	2	New device is not patched to latest version and can therefore contain vulnerabilities on provisioning.	Policy on up-to-date software on provisioning.
	1,3	Malware leaking data.	Intrusion detection systems (IDS), malware detection, encryption of DAR.
Denial of Service	1,3	Connectivity or bandwidth attack - overload of requests in any direction.	Hardening, limiting allowed requests from one endpoint, limiting amount of resources needed for handling a not-authorized connection, firewalls, etc.
	1,2,3	Malware alters the system behaviour	Malware detection methods.
	1,3	Replay attack intended to disrupt or alter functionality.	Using e.g., session token to invalidate old messages.
	1,3	Routing attack disrupting data flows.	IDS and malware detection also on routing nodes.
Elevation of Privileges	1,3	A legitimate actor gains access of a resource without proper privileges.	Authorization using least privilege principle.
	1,3	A process running on the same device as another service gains access to e.g., memory or disk outside its intended scope.	Proper sand-boxing, parameter-bound checking, etc.

Table 7.1: A simplified threat model derived using STRIDE model

3) ***Organization:*** policies on actions to take when provisioning and decommissioning devices.
4) ***Infrastructure:*** identification, authentication and authorization of devices, services, users, integrity and encryption of Data in Motion (DIM) including forward/backward security, malware detection, audit log monitoring, intrusion detection systems (IDS).

When considering a large scale IIoT system as described in Section 7.2, requirements related to the infrastructure are likely to be the most challenging ones. Therefore, in the following we assess different countermeasures deduced from the threat model and related to infrastructure, and enumerate their related state of the art or best-practice solutions.

7.4.1 Identification

In Scenario 1 applied to an IIoT system, the trend-service must be able to identify the pump and flow-meter to find service end-points for receiving I/O-data for the devices. It is reasonable that actors communicating in any way must be able to deduce the identity of each other. In Scenario 2, the physical pump device is replaced, so there is a need to propagate changed identity to dependent actors, or update mapping between identities in different name-spaces so that the system as a whole retain its functionality.

Radio Frequency Identification (RFID) [33] is used as means for contact-less transfer of identity in several IIoT applications, e.g., in logistics. In network technology, MAC addresses may be used to uniquely identify an Ethernet card. In software technologies, Global Unique Identifiers (GUID) are often used for identifying different entities. Serial numbers and physical addresses could also be used for identification. An entity may hold several unique identities that are relevant for different actors. To interact, each actor must have knowledge of at least one of the other entity identities, and there must be a well-known and trusted method for translation between different name-spaces.

7.4.2 Authentication

None of the identification schemes described in the previous section provide proof of identity per se. RFID technology is for example vulnerable to both impersonation and Denial of Service (DoS) attacks [1]. Authentication is the method where an actor presents proof for a given identity, usually referred to as credentials. Therefore all actors (e.g., an I/O-data service for the pump, a software patch publisher, etc.) must be able to authenticate themselves. Furthermore, for actors that have to interact with each other, a common method

for authentication is needed.

A number of techniques exist for authentication, e.g., using a shared secret (password), digital certificates (x.509) and signatures such as RSA-PSS, DSA, BLS, bio-metrical measures (e.g., fingerprints), etc. Using a trusted third party for providing authentication is a way to enable actors to establish trust without prior knowledge of each other. Kerberos [31] is one such protocol for a secure authentication over a non-secure network using a trusted third part, where several implementations exists, using different combinations of cryptographic algorithms. OpenID is an open standard and protocol commonly used for enabling websites to authenticate users on the website with e.g., Google or Facebook as identity providers. Signatures from certificates with a common trusted root certificate is another way to provide authentication.

7.4.3 Authorization

Authorization is a method of connecting an identity with a set of privileges. In the case of Scenario 2, the new pump must be authorized to perform any action the old defective pump was able to e.g., reading I/O data from flow sensor, at the same time as the defective pump must have all its privileges revoked.

Granting and validating privileges of an actor can be done in several ways such as: *1)* Identity based authority, meaning that the owner of the resource the actors wants to access, keeps a record of identities paired with privileges e.g., Access Control Lists (ACL); *2)* Attribute Based Access Control (ABAC) where attributes of an actor are used in deciding authority; *3)* Role-based Access Control (RBAC), the owner of the resource allocates certain privileges to specific roles, and there is a way to deduce a role from an identity; *4)* Information flow focused methods based on sensitivity levels of information and clearance levels of actors.

Available technical solutions include OAuth [16], an open standard for delegating authorization for HTTP-based applications. Extensible Access Control Markup Language (XACML) is a standard and framework that can be used for describing access control policies using both ABAC and RBAC. PERMIS [5] is a framework focusing on RBAC, but using a certificates based infrastructure to define roles and privileges.

7.4.4 Integrity

Integrity of data is the characteristic proving that the data have not been maliciously or accidentally changed or destroyed [34]. Data needs to be checked for integrity, both when reading from storage to protect against tampering of Data at Rest (i.e. DAR) and when receiving data over a network (i.e., DIM). Using check-sums and Message Authentication Code (MAC) are standard methods for ensuring integrity of data.

In Scenario 3, the integrity of the data flow from device to manufacturer is crucial, as tampering of data could prevent urgent patches being applied. Integrity of the patch itself is also very important, as a tampered software update would possibly inject malware into the device. Software ID (SWID) tagging as described in ISO/IEC 19770-2 [21] is one technique to assure software update integrity.

In Scenario 2, the authenticity of the new pump-type can be questioned. It would be desirable to detect a counterfeit, as it could contain malware, underperform, etc. To prove that the device software is authentic it might provide a digital signature from the vendor based on a certificate originating from a well known certification authority.

7.4.5 Encryption

Encryption is a method of rendering data unreadable to anyone not holding a deciphering key. Symmetric encryption methods, such as AES, use the same key for encryption and decryption. In asymmetric encryption methods, such as RSA, the key for encryption and decryption differs and this is the basis for any public-key technique where the key used for encryption is made public and the key for decryption is secret. Asymmetric encryption enables secure communication without previously shared secrets. The strength of any encryption algorithm is related to the length of the deciphering key.

If the trend service described in Scenario 1 is accessing sensitive data, it must be protected from unintended viewers both at rest (i.e., DAR) and in motion (i.e., DIM). The transfer of a software patch between a publisher and device, as described in the previous section related to Scenario 3, should also be protected from eavesdropping. An attacker could, e.g., use the software to perform a binary analysis of it as a part of preparing a future attack.

To protect DIM it must be decided in which layer the protection should be implemented (e.g., link-layer, network, transport, or application). Securing communication only at the lowest levels might work well for some applications,

but may not provide granular enough security controls for all applications. For example if using a message broker to handle communication, sensitive data only intended for the receiver of the message must be encrypted before reaching the broker. A combination of network/transport and application layer protection could be applicable in such cases. Common state-of-the-art protocols for protection of DIM are: IPSec, Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS), Wireless Transport Layer Security (WTLS). For situations where the intended receiver is not know, or there are several receivers, Attribute Based Encryption (ABE) [14] can be used. Using this technique a publisher is decoupled from a subscriber using a trusted key-host.

7.4.6 Audit log

An audit log is a record or set of records containing timestamped actions of predefined types, typically security related events such as failed login attempts or access to sensitive data. What should be logged depends on the security policy of the organisation. Audit logging is needed to perform forensic analysis and prove repudiation. Monitoring audit logs is also a way to detect attack attempts, as a wide range of attack vectors may be utilized before an attack is successful. The audit logs are themselves therefore possible attack-targets and must therefore be protected at storage and transferred using secure channels. For example, in Scenario 3, the patch publisher might claim that the software patch is sent to the pump device, while on the other hand the pump device might claim not to have received any patch. Audit logging is used to prove or disprove these competing claims.

7.4.7 Malware detection

Malware stands for for malicious software, meaning software performing actions not desired in the system, e.g., leaking data, altering device behaviour, using up local or remote resources, etc. Several of the deduced threats and countermeasures are related to a device or service malfunctioning. The patch being applied in Scenario 3 might contain malware, the pump installed in Scenario 2 might contain malware, any of the I/O-data being displayed in the trend in Scenario 1 might deviate from the real values due to a malfunction, etc. There are several possible root causes for a defective behavior of a device (i.e., mechanical or electrical error, network disturbances, etc.) and the method for detecting the fault differs based on the root cause.

Methods for detecting malware include trusted boot, software attestation,

IDS, application white-listing and anti-virus software. Trusted boot is used to assert that any software loaded during the boot is the expected one (e.g., an operating system). Attestation is a method were the executing software of a device is validated often using a challenge-response method. Both self-attestation and remote attestation is possible. Trusted Platform Module (TPM) [6] is a hardware module that among other things can support trusted boot and self-attestation. SMART [9] is an example of an architecture for providing attestation for resource constrained devices. IDS [15] is a mechanism for monitoring activities in a system, compare with the past behavior and known attack patterns, and report upon finding anomalies.

7.5 Challenges and future directions

In Section 7.4 previously introduced scenarios (see Section 7.3) are generalized to a number of basic security requirements with respect to the infrastructure of a large-scale IIoT system. In this section we discuss some of the challenges for such an infrastructure in an IIoT context.

7.5.1 Interoperability

For the previously discussed security related requirements, there are several applicable state of the art solutions. Different protocols for communication, authorization and authentication exist, as well as many algorithms for encryption and information integrity. Considering a large IIoT, there will be devices and services implemented using competing technologies that must be able to communicate within the same system. Frustaci et al. [12] provide a classification of commonly used IoT protocols at physical, network and application layer, including each protocols' security issues and related solutions. To combine several exiting protocols and standards currently in use in industrial applications (e.g., OPC UA, PROFINET, MODBUS, etc.) it becomes a challenging task to enable basic interoperability with regards to communication. It does not seem feasible to limit the communication capabilities in an IIoT system to a few interoperable protocol implementations, as it will put unreasonable constraints on the devices. Instead a unifying methodology that allow cross-technology communication is required.

Let us consider Scenario 1 described in Section 7.3. I/O-data from the sensor could e.g. be accessible from an OPC UA server, I/O-data from the pump-device could be accessible from a MQTT message broker running on an edge device. For every such type of data-source that must be handled by the trend-service, a significant amount of implementation work is required.

Furthermore, it will be virtually impossible to know at design-time which types of data-sources the trend service must be able to support.

Looking at the available architectures it is not clear how to achieve interoperability between competing technologies, or technologies never intended to be interoperable when designed. As a solution, RAMI4.0 requires that all entities must have an administrative shell and component manager that exposes services and data in a very uniform way to be part of an Industry 4.0 system.

Using the layered databus architecture pattern is one way of handling communication interoperability. Such architecture only requires for each logical layer the existence of a common data model allowing the entities within that layer to communicate. Between each layer there is a databus gateway, enabling flow between layers. For interoperability between layers there is a need for adapters in gateways to translate between the data-models [19]. The idea of the layered databus is however not to allow free communication between arbitrary endpoints. To allow secure communication in this context seems to be quite difficult, as data will be transformed at every gateway.

Yet another way of looking at solutions for interoperability between actors without prior knowledge of each others technological stack is to use an App-centric view. Assume that a service S_1 running on a device D_1 wants to access a service S_2, but they are implemented using competing technologies. Now, if D_1 can execute concurrent services (e.g., using docker containers) and there is a well-defined secure method for local communication between services on D_1 (e.g., an internal message broker), then it would be enough that a service is created such that it can execute on D_1 which reads the data from S_2 and then post the data on the internal message broker. It should be noted that any of the suggested solutions will require some predefined functionality for fetching meta-data about a device or a service.

7.5.2 Management of privileges, identity mappings and data classifications

In any system there is a need to administer different characteristics for included actors.

- Manage Identities (e.g., add and remove users, map device IDs to location and functions, etc.).
- Manage Privileges (Which user/service/device is allowed to read specific information or execute an action.).

- Classification of data (Whether the data should be encrypted at rest, in motion, is it sensitive or not, does it contain information that requires it to be stored in accordance with e.g., GDPR, etc.).
- Maintenance Scheduling (e.g., when to replace the pump in Scenario 2, when to apply the patch in Scenario 3, an so on).

Even for a quite small amount of actors this can be a tedious task. For a large scale IIoT system, the number of actors is huge and their relationship may not be predefined. Such administration might be complex and time-consuming. Therefore, for a technology only requiring high-level configuration and promising autonomy at the lower levels, the management at the lower level must somehow be automated. For example, in scenario 2, there could be logic based on proximity detecting the new pump device and assigning it roles and privileges accordingly.

One has to keep in mind that the best-practice for authorization is the principle of least privilege. This principle states that an actor will only be granted privileges needed to perform its intended function. In an IIoT system it will be difficult to beforehand deduce what the least privileges are, potentially forcing higher privileges being granted than actually required. This could lead to a conflict with the least-privilege principle. Solving the issue of automatic management of identities, privileges, etc., remains an open question.

For some situations there might not even exist an omnipotent actor able to decide on privileges. In the case of a smart city with autonomous vehicles and smart traffic control it is reasonable that, to ensure traffic safety, a vehicle from another city or country should be allowed to communicate with other vehicles and infrastructure without prior knowledge or registration in this specific system. Smart contracts utilizing block-chains could be a way forward for preserving reliability in such scenarios [8], as well as zero-knowledge proof [29].

7.5.3 Fault and anomaly Detection

There is always an amount of uncertainty when evaluating the state of the real world. Any sensing device has a tolerance-level indicating how exact the sensor is, and for any actuating device the effect of the actuation will be based on a model of reality, which never is perfect. In scenario 1, an undetected anomaly being presented to the operator could lead to erroneous decisions being made. Malware introduced in the example system, e.g., by a malicious software update introduced in scenario 3, could lead to loss of control, as well

as information leakage.

Common ways to decrease the level of uncertainty is to use e.g., secondary data-sources, or to compare model data with sensor data. These techniques could possibly be an extended form of detecting malfunctioning devices, as well as a methods for intrusion detection, which currently are a growing area of research [13].

Attestation as a method to detect malware at high-end devices is a very promising technique, but, as described by Sadeghi el al. [28], current solutions do not scale well, especially not for low-end devices. To find applicable solutions for IIoT including attestation of large amounts of devices in parallel, so called *swarm-attestation*, is still an open field for research.

The standard IDS is focused on probing network traffic to monitor and detect anomalies or predefined attack patterns and report findings to a security function (i.e., a human or a machine) so that the anomaly can be classified. These systems are well suited for detecting suspicious patterns in communication, but will have increasing difficulties in finding anomalies in data content, as the data itself often will be encrypted in an IIoT system.

Another issue of increasing importance for the IDS is knowing which traffic to monitor. In a heterogeneous IIoT system there will be devices communicating using any number of diverse wired and wireless technologies. The fifth generation telecommunication standard (i.e., 5G) is believed to be an enabling technology for wireless cross-component communication in IIoT systems [23]. In a scenario where communication is done partially using wireless communication capabilities such as 5G, the traditional IDS with trusted nodes inspecting passing traffic will not work, as much of the traffic will not pass the trusted node. For such scenarios, IDS in an ad-hoc mobile network could possibly be used [15]. Such an IDS is based on collaborating agents being deployed on many nodes, using joint status and voting to decide on anomaly detection.

Monitoring audit logs is a way for early detection of attempted intrusions, as the logs will contain information on failed access attempts. It would be possible to use these as means for intrusion detection in an IIoT system. To be useful, the detection system must be able to remotely access and monitor audit logs for a wide range of devices and services and automatically detect unexpected patterns. Security Information and Event Management (SIEM) [3] is a technology that focus on storage and analysis of audit logs. How well existing SIEM solutions perform in and scale to a heterogeneous IIoT system must be further investigated.

7.5.4 Emerging threats and technologies

The secure operation of a device is limited to the capabilities available in the device, as implemented by the manufacturer. A device may be secure at provisioning, but its continuous state with regards to security is dependent on its possibility to adapt to emerging threats and technologies. Considering that the average lifetime for machine equipment is expressed in decades [10], it will be impossible to equip devices with hardware capabilities that will match requirements for state of the art in security lasting the whole expected lifetime. It is however essential that the software for the device is kept up to date with current threats and adapts to emerging technologies as long as possible. Secure patch management and methods for assessing the status of a device software with regards to security functions is therefore of great importance to handle the risks introduced in scenario 3 as well as keeping the device software up to date. When replacing hardware as described in scenario 2, it is important to make sure that the new device is able to conform not only the functional requirements of the system, but also with regards to current cybersecurity state of the art technology.

When adding IIoT features in a brownfield system, e.g., exposing information to the Cloud, this is usually done by putting a gateway device between the information producer and consumer. The gateway will provide the security functionality required for devices that it is servicing [18]. A similar approach can possibly be used for keeping out-dated IIoT devices secure. Such a solution would require that all communication from the IIoT-device can be relayed to the gateway/proxy. For devices with wireless networking capabilities, for example built-in mobile communication chips, this solution may not be straightforward, depending on the device capabilities. In general, handling emerging threats and technologies for resource-constrained devices is very much an open issue.

7.6 Related Work

Frustaci et al. [12], provide a thorough analysis of current state of the art for securing IoT devices and data, as well as an evaluation of identified critical security issues related to IoT. The focus is on resource-constrained devices for consumer use, assuming that those devices will rely on “built-in security”. In some aspects this is clearly the case also for industrial applications, as devices both for industrial and consumer applications will be constrained with regards to computational and storage capacity. However, this does not hold for software. As we have indicated in this paper, there is a clear requirement

on device software to be patched to counteract emerging threats and discovered vulnerabilities. There is also an emphasis on the physical layer bringing the highest risk to the IoT-system. In contrast, most of the physical risk to the devices are not considered in this paper. For industrial applications there is usually already a layer of physical security with fences, locked doors, access control, etc. This may not hold for all industrial applications, e.g., geographically distributed processes such as a gas or oil pipeline.

Chiang et al. [7] discuss several fundamental challenges, using traditional cloud technology within the emerging IoT, and provide arguments for using fog nodes to counteract some of these challenges, e.g., related to latency requirements, bandwidth constraints, intermittent connectivity, etc. The focus is IoT in broad terms, including both consumer and industrial applications. A number of security related challenges are discussed, some of which are described more in depth in our work, e.g., keeping security credentials and software up to date and protecting resource-constrained devices. In this paper we have consistently suggested that services will be spread out through the thing-to-cloud continuum, thereby including fog nodes. This will also be true for security related services, such as IDS, remote attestation, etc. Chiang et al. also acknowledge that fog technology introduces new security challenges, as such nodes are as diverse and distributed as IoT devices, as opposed to cloud which in general operates in a protected environment.

Sadeghi et al. [28] provide an overview of security challenges for Cyber-Physical Production Systems (CPPS). Integrity of device software is discussed as one of the challenges, with attestation of integrity needed to be performed by a trusted entity. Secure IoT management is also discussed as one of the important areas for future research. In this context the notion of "pairing" of devices are used, as done with PIN-codes on Bluetooth devices (i.e., pairing headset with cell-phone), which could be an interesting way to handle inter-device identification and authorization with minimal human interaction.

7.7 Conclusions

In this paper we focus on the emerging IIoT systems being a combination of Industrial Automated Control Systems and Internet technology. In such systems, smart CPS devices and services are applied throughout the device-to-cloud continuum with heterogeneous technologies and multiple stakeholder that put high requirements on the underlying infrastructure. We have discussed a number of challenges in such a setup from a cybersecurity perspective using an example of a flow-control loop process. For such an example

we describe three scenarios and apply a STRIDE threat model to deduce and discuss cybersecurity challenges within an IIoT perspective.

As future work, we aim to analyze emerging IIoT systems in more detail, focusing on cybersecurity aspects identified in this paper in order to provide required solutions. The goal is to analyze solutions applicable to a truly modular infrastructure for cybersecurity that scale well with regards to large-scale IIoT systems.

Acknowledgements

This work is supported by ABB, the industrial postgraduate school Automation Region Research Academy (ARRAY), and SAFSEC-CPS, projects funded by The Knowledge Foundation. Additional support is provided by Serendipity, a project funded by The Swedish Foundation for Strategic Research. The authors would like to acknowledge Mikael Rudin and Tomas Lindström for valuable discussions and feedback when writing this paper, as well as anonymous reviewers for their comments.

Bibliography

[1] G. Avoine. *Encyclopedia of Cryptography and Security: RFID Security*, pages 1044–1045. Springer US, Boston, MA, 2011.

[2] R. Baheti and H. Gill. Cyber-pysical Systems. *The impact of control technology*, 12(1):161–166, 2011.

[3] S. Bhatt, P. K. Manadhata, and L. Zomlot. The Operational Role of Security Information and Event Management Systems. *IEEE Security & Privacy*, (October), 2014.

[4] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson. The industrial internet of things (IIoT): An analysis framework. *Computers in Industry*, 101:1–12, June 2018.

[5] D. W. Chadwick and A. Otenko. The PERMIS X.509 role based privilege management infrastructure. *Future Generation Computer Systems*, 19(2):277–289, 2003.

[6] D. Challener. *Encyclopedia of Cryptography and Security: TPM*, pages 1308–1310. Springer US, Boston, MA, 2011.

[7] M. Chiang and T. Zhang. Fog and IoT: An Overview of Research Opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, 2016.

[8] K. Christidis and M. Devetsikiotis. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access*, 4:2292–2303, 2016.

[9] K. E. Defrawy, A. Francillon, D. Perito, and G. Tsudik. SMART: Secure and Minimal Architecture for (Establishing a Dynamic) Root of Trust. *NDSS*, 2012.

[10] A. A. Erumban. Lifetimes of machinery and equipment: Evidence from dutch manufacturing. *Review of Income and Wealth*, 54(2), jun 2008.

[11] FIRST. Common Vulnerability Scoring System. `https://www.first.org/cvss/`, 2019. [Online; accessed 29-may-2019].

[12] M. Frustaci, P. Pace, G. Aloi, and G. Fortino. Evaluating critical security issues of the IoT world: Present and future challenges. *IEEE Internet of Things Journal*, 5:2483–2495, 2018.

[13] J. Giraldo et al. A Survey of Physics-Based Attack Detection in Cyber-Physical Systems. *ACM Computing Surveys*, 51(4), 2018.

[14] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based Encryption for Fine-grained Access Control of Encrypted Data. In *13th ACM Conference on Computer and Communications Security*. ACM, 2006.

[15] Q. Gu. *Encyclopedia of Cryptography and Security: Intrusion Detection in Ad Hoc Networks*, pages 620–623. Springer US, Boston, MA, 2011.

[16] D. Hardt. The OAuth 2.0 Authorization Framework. Internet Requests for Comments, October 2012.

[17] M. Hermann, T. Pentek, and B. Otto. Design principles for industrie 4.0 scenarios. In *Proceedings of the Hawaii International Conference on System Sciences*, volume 2016-March, pages 3928–3937. IEEE, 2016.

[18] IIC. The Industrial Internet of Things Volume G4 : Security Framework. Technical report, Industrial Internet Consortium, 2016.

[19] IIC. The Industrial Internet of Things Volume G1: Reference Architecture. Technical Report November, Industrial Internet Consortium, 2017.

[20] International Electrotechnical Commission. Smart Manufacturing - Reference Architecture Module Industry 4.0 (RAMI4.0), 2016.

[21] ISO IEC. ISO/IEC 19770-2:2015 IT Asset Management Part 2: Software Identification tag. Technical report, ISO/IEC, 2015.

[22] R. Kissel. *Glossary of key information security terms, Revision 2.* U.S. Dept. of Commerce, National Institute of Standards and Technology, 2013.

[23] S. Li, L. D. Xu, and S. Zhao. 5G Internet of Things: A survey. *Journal of Industrial Information Integration*, 10:1–9, 2018.

[24] W. Madsen. *Trust in Cyberspace.* National Academies Press, 1999.

[25] Microsoft. The STRIDE Threat Model. `https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)`, 2005. [Online; accessed 5-march-2019].

[26] Microsoft. Microsoft Azure IoT Reference Architecture. `https://aka.ms/iotrefarchitecture`, 2018. [Online; accessed 29-may-2019].

[27] S. Mumtaz et al. Massive Internet of Things for Industrial Applications: Addressing Wireless IIoT Connectivity Challenges and Ecosystem Fragmentation. *IEEE Ind. Elec. Magazine*, 11(1), 2017.

[28] A.-R. Sadeghi, C. Wachsmann, and M. Waidner. Security and privacy challenges in industrial internet of things. *Proceedings of the 52nd Annual Design Automation Conference on - DAC '15*, pages 1–6, 2015.

[29] B. Schoenmakers. *Encyclopedia of Cryptography and Security: Zero-Knowledge*, pages 1401–1403. Springer US, Boston, MA, 2011.

[30] R. Shahan and B. Lamos. Internet of Things (IoT) security architecture. `https://docs.microsoft.com/en-us/azure/iot-fundamentals/iot-security-architecture`, 2018. [Online; accessed 29-may-2019].

[31] J. G. Steiner, C. Neuman, and J. I. Schiller. Kerberos: An Authentication Service for Open Network Systems. *WTEC 1988: Proceedings of the USENIX Winter 1988 Technical Conference*, pages 191–202, 1988.

[32] T. UcedaVelez and M. M. Morana. *Risk Centric Threat Modeling: process for attack simulation and threat analysis.* Wiley & Sons publishing, 2015.

[33] R. Want. An introduction to RFID technology. *IEEE Pervasive Computing*, 5:25–33, 01 2006.

[34] M. H. Weik. *Computer Science and Communications Dictionary: Data integrity*, pages 350–350. Springer US, Boston, MA, 2001.

Article B

Chapter 8

Article B: Applicability of the IEC 62443 standard in Industry 4.0 / IIoT

Björn Leander, Aida Čaušević, Hans Hansson
In proceedings of the 14th International Conference on Availability, Reliability and Security (ARES), Canterbury, United Kingdom, August 2019

Abstract

Today's industrial automation systems are undergoing a digital transformation that implies a shift towards the Internet of Things (IoT), leading to the Industrial Internet of Things (IIoT) paradigm. Existing Industrial Automated Control Systems (IACS), enriched with a potentially large number of IoT devices are expected to make systems more efficient, flexible, provide intelligence, and ultimately enable autonomous control. In general, the majority of such systems come with high level of criticality that calls for well-established methods and approaches when achieving cybersecurity, preferably prescribed by a standard.

IEC 62443 is an industrial standard that provides procedures to manage risks related to cybersecurity threats in IACS. Given the new IIoT paradigm, it is likely that existing standards are not sufficiently aligned with the challenges related to developing and maintaining cybersecurity in such systems. In this paper we review the applicability of the IEC 62443 standard in IIoT contexts and discuss potential challenges the process owners might encounter.

Our analysis underlines that some areas within the standard could prove difficult to reach compliance with. In particular, handling of cross zone communication and software updates require additional guidance.

8.1 Introduction

Industrial Automation and Control Systems (IACS) are used for operating a wide range of industrial applications, including critical infrastructure. An emerging trend within IACS is the Industrial Internet of Things (IIoT), being driven by the fourth industrial revolution (Industry 4.0). According to Industrial Electrotechnical Commission (IEC) [6], a fundamental purpose of Industry 4.0 is to enable cooperation and collaboration between devices. More specifically, the aim of IIoT is to enable optimization, cost-savings, and new business opportunities in different domains. It is expected that IIoT will introduce significant advances in optimizing decision-making, operations and collaborations among a large number of increasingly autonomous control systems [8].

IEC 62443 [7] is an industry standard that describes ways to handle cybersecurity threats in IACS. The standard has been developed with the classical automation pyramid in mind. With the emergence of IIoT, this architecture is no longer the norm, and the development has accelerated an already ongoing convergence between Operation Technology (OT) and Information Technology (IT) that results in an increase of the attack surface of IACS. There is an apparent risk that the introduction of IIoT makes parts of the standards outdated.

The main purpose of this paper is to assess the IEC 62443 standard from an IIoT perspective, and discuss a number of issues that process owners will face when trying to keep compliance to the standard while adapting to the reality of an increasing number of IIoT devices being part of the system. To make the work more readable we include a rather simple description of an automation architecture in both a traditional IACS and an IIoT set up, to which we relate our findings.

The paper is organised as follows. Section 8.2 introduces necessary background and defines concepts used in this paper. In Section 8.3 the current state of the IEC 62443 standard is described together with the IACS reference model. Section 8.4 presents a simplified architecture for an IIoT system, and based on that we analyse the IEC 62443 standard, and provide a discussion on challenges when trying to reach compliance to the standard in such a system. The contributions of the work are recalled in Section 8.5, together with suggestions for the future research.

8.2 Background

An IACS is defined as the system of hardware, software, personnel and policies involved in operation of an industrial process and that can affect its operation with regards to safety, security and reliability [7]. IACS are responsible for controlling and monitoring a wide range of different types of physical processes, ranging from chemical industries, power plants, manufacturing, etc. Many of these systems are of vital importance for supplying basic functionality to society, such as electricity and clean water. Failure of systems providing critical infrastructure services can have severe effects, both economical and environmental, and their protection is therefore of great importance. For many industry segments there are laws regulating how this protection must be implemented. For example, plants delivering power to the North American power grid are required to fulfill the NERC CIP standard [15].

Cybersecurity is the protection of a computer system from unauthorized actors possibility to steal or alter information in the system, disrupt or alter behaviour of a function or perform an unauthorized action [11].

The IEC 62443 is the de facto standard for cybersecurity in industrial control systems, as the only one being applied internationally and cross-industry [12]. It is defined by the IEC in cooperation with International Society for Automation (ISA). IEC 62443 has parts being under development, but it is still widely used by industry, and also forms a base for certification, e.g. the Embedded Device Security Assurance (EDSA) certification [10]. An IACS owner can use the described methods to keep its system at a desired level of security, and also require that service providers and manufacturers of the components used in the IACS follow the principles and adheres to a certain security level for their delivery. In this way the IEC 62443 is a source of common understanding of cybersecurity related issues for IACS owners, component developers, and service providers.

In the traditional IACS there used to be a clear separation between the OT network and the IT network. The OT network containing the devices and services directly concerned with controlling the physical process, was usually physically separated from the IT network, that contained e.g., the organization office network. There is an ongoing convergence between the IT and OT network, with the introduction of IT technology in the OT network, and a growing amount of interconnections between IT and OT networks, e.g., remote access from IT clients to OT functions and the usage of standard IT components in OT systems. This convergence of technologies implemented with different objectives with regards to security [12] is exposing IACS to po-

tentially new cybersecurity threats. The attack on the Ukrainian power grid in December 2015, is one such an example, where attackers were able to compromise and disrupt power distribution [13], affecting approximately 250.000 Ukrainian citizens.

The Industrial Internet or Industry 4.0 is an ongoing trend in the world of industrial automation. Some of the promises of Industry 4.0 are:

- Autonomous collaboration between technical assets, minimizing the need for low level configuration.
- Advanced analysis of large amounts of data allowing better business decisions.
- Support for novel business models, such as Factory as a Service.

Internet technology is being applied in IACS systems, and specifically IoT devices and services being adopted to or developed specifically for use in industrial applications. IIoT has a multitude of definitions, but in this paper we will use the following definition, inspired by Boyes et al. [2]: an IIoT is assumed to be comprised of devices and services spread over a thing-to-cloud continuum, with each device able to be composed of several devices. Devices may have related information spread throughout several services, and for each device there may be multiple stakeholders both within and outside the IACS owner organisation. The objective of the IIoT is to optimise the overall value that the IACS deliver, including e.g., product or service quality, productivity, labour costs and resource allocations. In smart manufacturing, the product being manufactured is also part of the IIoT, directing the process-steps it flows through with actions that must be executed to complete its manufacturing process.

8.3 IEC 62443 - Current state

IEC 62443 consist of a number of documents describing different aspects of implementing and maintaining security to a well defined level within an IACS. The standard is split into four main groups, with several documents in each group:

- IEC 62443-1-X General, contains documents for defining concepts, terminology, use cases, etc.
- IEC 62443-2-X Policies and procedures, contains e.g., secure patch management and security program requirements.

- IEC 62443-3-X System level requirements, system risk assessment, etc.
- IEC 62443-4-X Component level requirements, including component development requirements.

In this paper we will look at published documents of the standard available from the IEC library. At the time of writing this includes 1-1, 2-1, 2-3, 2-4, 3-1, 3-3, 4-1 and 4-2.

The IEC 62443 standard in general provides requirements that must be fulfilled, but does not suggest measures for evaluating implementation of these requirements. There is no clear guidelines in process of assuring that the requirements are met, which makes a lot of the work with assigning levels of security and assessing countermeasures into subjective tasks for the implementing organisation. This characteristic makes the standard useful also when new technologies are introduced, but partly impede the possibility of stating compliance without subjective judgement.

Risk tolerance level is one key aspect defining the risk an organisation can accept for a specific IACS. Several different response strategies can be applied to a risk:

- Change design to remove the risk;
- Reduce the risk;
- Accept the risk;
- Transfer the risk, e.g., insurances or outsourcing of function.

Cybersecurity Management Systems (CSMS) includes e.g., programs to continuously reassess risks. Security Levels are created to classify groups of assets, with regards to security zones. For each security zone a target security level $SL(target)$ is assigned. The $SL(target)$ is usually the outcome of a risk assessment of that zone. $SL(target)$ describes the effectiveness that applied countermeasures must reach to properly secure the zone. The achieved security level $SL(achieved)$ of a zone is a dynamic property that typically degrades with time, as emerging threats and evolving technologies make existing countermeasures relatively less secure, unless maintenance and upgrade procedures are followed. $SL(capability)$ is the security level a specific countermeasure or device/system can provide to a security zone.

The goal is that for any given time $SL(acheived) \geq SL(target)$, for each security zone defined in the system. A security level life-cycle aims to continuously fulfill this goal, using recurring reassessments and specific assessments

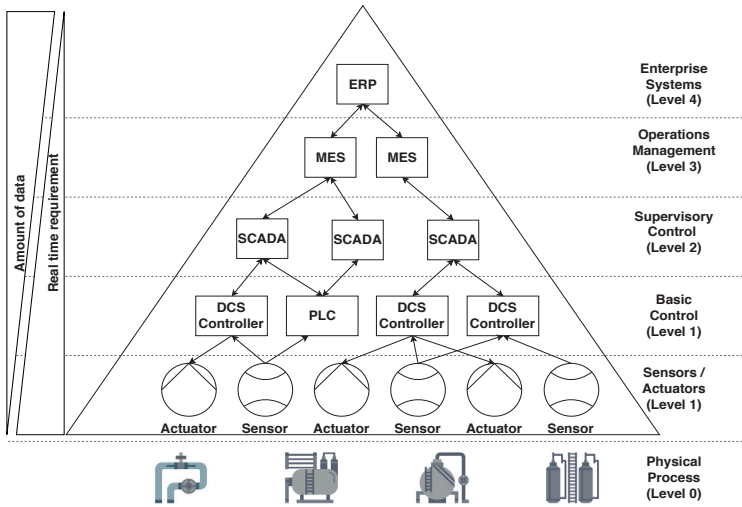


Figure 8.1: Traditional automation pyramid based on PERA

for security related system changes, e.g., process change, new vulnerability detected, software patch of devices.

The IACS reference model used in the standard is 5-tier, influenced by the Purdue Enterprise Reference Architecture (PERA) [22], illustrated in Figure 9.1a.

Layer 1-2 typically comprise the OT network, usually being split into several security zones based on criticality, layer 3-4 comprise the IT network. As can be seen the different layers directly interact only through hierarchy. Lower levels typically have real-time constraints, but the higher the level, the longer the cycles become. For Enterprise Resource Planning systems (ERP) reaction on data may be in terms of weeks or months. The amount of data being collected and concentrated per level is reversed, the higher the level, the more data is used for the processing logic.

8.3.1 Security program for IACS service provider and IACS owner

IEC 62443-2-1 Ed. 1 and 62443-2-4 contains guidance on the content and development for a CSMS for an organisation owning or providing service to an IACS. The standards mainly consist of policies and procedures, that shall be part of the CSMS, and suggestions on how these could be developed.

The elements of the CSMS with regards to IACS owner is divided into three

main categories, the first one focusing on risk analysis, the second (and largest) one focusing on addressing risks, and the third one addresses fulfillment and continuous improvement of the CSMS.

The elements focusing on risk analysis provides requirements on e.g., that a risk assessment methodology must be selected, that a risk assessment using that methodology should be executed and documented by trained personnel and that there should be a strategy for reassessment.

The elements focusing on addressing risk contains requirements on policies, organization, selected security countermeasures, document management, incident handling, etc.

The elements focusing on fulfillment and improvement of the CSMS contains requirements on how to perform recurring audits of the organisation, and how to evaluate and introduce changes of the CSMS.

IACS service providers are separated into two categories: integration service providers and maintenance service providers. The requirements as defined for CSMS for IACS service providers are formulated slightly different compared to those of an IACS owner, as the focus is on what capability the service provider can deliver in relation to the IACS. The Capability Maturity Model Integration for services (CMMI-SVC) [4] is adapted to the standard as a measure for service provider maturity with regards to compliance with the standard.

8.3.2 Secure Patch Management

Secure patch management is an issue of great importance in an IACS, as software goes out of date, bugs are fixed, potentially functionality is added. At the same time, introduction of non-operable or malicious software poses a great threat to such a system.

IEC 62443-2-3 is the part of the standard that provides guidance on secure patch management. All assets must be monitored with regards to current versions and available patches, installed and verified in a test-system, create backups of original system before applying patch, and possibly halt operations while applying patch. Assets may reach a point in time when they are no longer supported by the product supplier, i.e., software/asset obsolescence. In such cases new patches for the asset will not be released regardless of any vulnerabilities or bugs discovered.

With the full patch management process both by the vendor and by the asset

owner, a software patch has a life-cycle containing several states, including testing, approving and releasing from product supplier perspective to internal test, authorization and internal release by asset owner (i.e., 11 steps according to the standard).

The standard supplies a set of recommended requirements with regards to patch management for both the IACS owner and IACS product supplier. For the IACS owner the key issue is to keep an inventory of all updatable assets containing their current versions, latest available patch versions and status, regularly revise that list and apply patches after performing internal tests. For product supplier the requirements include supplying information on patch availability and applicability, warn customer in advance of "end-of-life" for product, etc.

The standard argues for any IACS owner and IACS product supplier to implement a patch management process to facilitate these requirements.

8.3.3 Security technologies for IACS

IEC 62443-3-1 provides an assessment of various cybersecurity tools, mitigation counter-measures, and technologies that can be used in IACS, followed by guidance on usage and known weaknesses of existing methods.

Authorization and authentication are two of the main areas being covered, discussing Role Based Acces Control (RBAC) as one useful, but not widely used method. The main weakness is that current RBAC systems in general are tied to specific technology stacks, such as COTS OS. IACS commonly include specialized devices that do not have this support by default, thus require development of interfaces against the (various) RBAC system(s). Furthermore, a centralized RBAC system would require any device to be covered to have access to a central server, making the operation of the IACS dependent on the health of the corporate network.

Network firewalls are discussed as an important tool for perimeter protection, including SW and HW firewalls, different filtering strategies, log monitoring, etc.

Symmetric Encryption is discussed, and noted not being commonly used in the IACS environment, as the control networks are seen as operating in physically secure zones. However, for traffic crossing unsecure networks, encryption of data is encouraged.

Public key (assymetric) encryption is seen as an important means of exchang-

ing symmetric keys, but is in general too resource consuming to be used in time-critical devices. Man-in-the middle attacks can be successfully launched against public key encryption methods, unless authenticity of communicating parties are validated by certificates.

Audit log monitoring is described as being an important method of detecting intrusion attempts. Focus is mainly on servers e.g., windows server machines, for which there exist centralized audit log methods.

Intrusion Detection Systems (IDSs) come in two flavors: Network IDS (NIDS) and Host IDS (HIDS). NIDS is most commonly deployed as a separate device, e.g., connected to a mirroring port on a network router or integrated in a router or firewall. NIDS checks all network data for either known attack-patterns or unexpected behavior. HIDS is installed as software on a host and can check the logs, network traffic and file-system for indications of completed or ongoing intrusions. A special variant of IDS also prevents an intrusion attempt by, e.g., blocking network traffic related to a detected intrusion attempt. There are several drawbacks of IDS, mainly related to the cost of applying to all sub-nets and hosts, cost of monitoring and cost of handling false positives.

Vulnerability scanners provide means of hardening the system, and can be used to detect: security policy deviations, bad configurations and software flaws. Typically these kinds of scans should be performed when re-assessing $SL(acheived)$ for an IACS. However, the scan itself can have a negative impact on the performance of the IACS, implying that the scan should ideally be performed in a lab-environment first to assess that the impact of the scan will not interfere with regular operations. Alternatively a vulnerability scan could be performed during a planned maintenance halt of the process.

Host Configuration Management (HCM) tools can be used to remotely edit default host configurations with regards to available software, as well as user access. In IACS this is not widely used, due to the lack of standardization of such systems with regards to the diversity of hosts.

Operating Systems are discussed in the standard, especially real-time operating systems (RTOS) are mentioned as having limited possibilities and abilities to counter cybersecurity threats. As e.g., DCS controllers and PLCs, in general execute on RTOS, these devices by their nature cannot function without network connectivity that makes them one of the most vulnerable parts of an IACS. These systems monitor and control real physical processes. The recommendation is to keep them on truly isolated networks, e.g., keep time-critical application traffic on a separate network. This will probably be true in early

adaptions to IIoT, with separation of real-time functionality for control and critical supervision from information collection with regards to analysis. In a longer perspective, IIoT devices could be part of an IACS as a real-time critical component, providing measurement feedback or process control.

8.3.4 System and Component security requirements and security levels

IEC 62243-3-3 and 4-2 describe system and component security requirements and security levels. It aims to provide requirements for the IACS, based on the seven foundation requirements (FR):

1. Identification and authentication control (IAC);
2. Use control (UC);
3. System integrity (SI);
4. Data confidentiality (DC);
5. Restricted data flow (RDF);
6. Timely response to events (TRE);
7. Resource availability (RA).

Each foundation requirement has a purpose statement, and defines four security levels (i.e., SL 1-4), for example for the data confidentiality FR the levels are defined as follows:

SL 1 Prevent the unauthorized disclosure of information via eavesdropping or casual exposure.

SL 2 Prevent the unauthorized disclosure of information to an entity actively searching for it using simple means with low resources, generic skills and low motivation.

SL 3 Prevent the unauthorized disclosure of information to an entity actively searching for it using sophisticated means with moderate resources, IACS specific skills and moderate motivation.

SL 4 Prevent the unauthorized disclosure of information to an entity actively searching for it using sophisticated means with extended resources, IACS specific skills and high motivation.

The higher the security level a control system reaches for a specific FR, the more persistent against an attack on that area the system should be. Typi-

cally SL 1 will protect against accidental leaks or low-motivation, low-resource attackers, whereas SL 4 will prevent attacks from a highly motivated and resourceful adversary. There is also an implicit SL 0, indicating no specific security protection necessary.

The FR are detailed in System Requirements (SR) and additional Requirement Enhancements (RE) which are related to the different security levels for the FR.

There is a special notion of essential functions, being required to maintain health, safety and environmental concerns. Essential functions cannot be negatively impacted by implementation of security requirements, e.g., accounts used for essential functions shall not be locked out, security functions shall not add significant delay on time-critical essential functions. This can lead to difficult trade-offs between availability and the other security objectives in the case of certain types of attacks and countermeasures.

In principal, when using these parts of the standard, the desired SL for a specific IACS or component is selected for each of the seven FR. This will lead to a number of SR and additional RE being applicable to the system. Each of these requirements must be fulfilled for the target SL to be reached. This also means that there is a (relatively) easy way to assess to which degree a certain SL is reached with regards to a specific FR.

In IEC62443-4-2 component requirements (CRs) are described, in a similar way as the system requirements. They are classified into four categories:

1. Software Application Requirements (SAR);
2. Embedded Device Requirements (EDR);
3. Host Device Requirements (HDR);
4. Network Device Requirements (NDR).

It is common that requirements are the same for all type of components, and therefore expressed only as general CRs.

1. Software application - one or more programs/services that interacts with the process or control system and are executing on an embedded or host device;
2. Embedded device - a specific purpose device with specialized hardware and firmware developed to fulfill that purpose. Typically the device is

directly or indirectly involved into monitoring or controlling a physical process and has real-time requirements to fulfill;

3. Host device - a general purpose device with capabilities of running several services, usually with an "open" OS, e.g., Windows or Linux;

4. Network device - a device that facilitate (or limits) data flow between devices, but does not directly interact with the process.

Common component Security Constraints comprise a number of constraints applicable to the components that may restrict the implementation of some security functions. Some examples of constraints are: essential functionality must be sustained, least privilege shall be used when appropriate, etc.

8.3.5 Secure development of IACS Components

IEC 62443-4-1 describes the best practices to follow when implementing IACS components. The standard is based partly on the Secure Development Lifecycle Assessment (SDLA) certification, as described by ISCI [9]. The document aims to support component suppliers. It is divided into eight main practices:

1. Secure Management;

2. Specification of security requirements;

3. Secure by Design;

4. Secure implementation;

5. Secure validation and testing;

6. Management of security related issues;

7. Security update management;

8. Security guidelines.

Each practice is described in detail, and divided into related requirements. The requirements are in the most cases described as a need for the development organization to have a process fulfilling specific goals, e.g., "Security requirements review (SR-5): A process shall be employed to ensure that security requirements are reviewed, updated as necessary and approved to ensure clarity, validity, alignment with the threat model, and their ability to be verified."

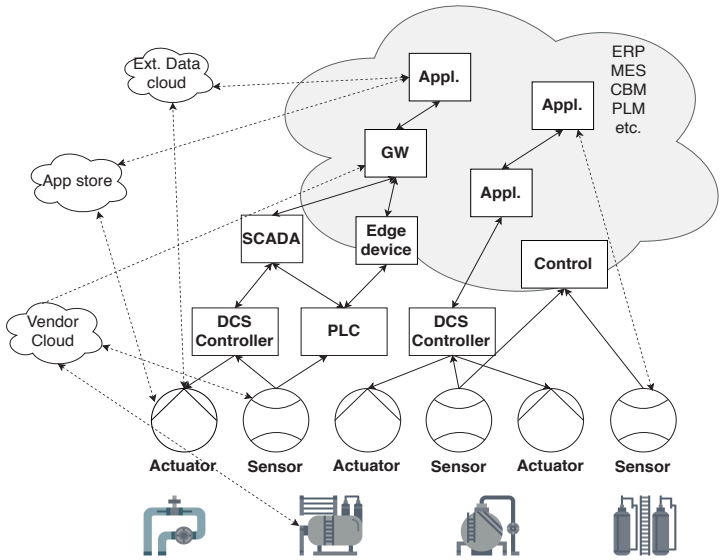


Figure 8.2: An example of an IIoT architecture [20]

If a product supplier is following these practices during the development lifecycle of an IACS component, the component will be able to comply to a specific SL over time, and will be secured by a defense in depth strategy.

Similarly as for the IACS service provider, the Capability Maturity Model Integration for development (CMMI-DEV) is used in the standard as a benchmark for a product supplier to indicate or self-assess to what degree the secure development processes for each practice are followed.

8.4 Assessment of IEC 62443 in relation to IIoT

As IIoT devices and services are being increasingly adopted into IACS systems they increase the potential attack surface of the system, as they often live at the edge of the network, i.e., communicate over the zone boundaries.

To not deteriorate the security characteristics of an IACS, as IIoT technology is introduced, it would be desirable to use the IEC 62443 standard to reassess the system, as well as for initial assessments if greenfield IACS implementations utilizing IIoT are provided.

8.4.1 IIoT systems - an architectural view

An asset in IIoT can be seen as the sum of all devices and services containing functionality or data for that asset. The system for an asset could be comprised by different services for current and historical process data, such as current control set-point, historical data for power consumption, data from a connected vibration sensor, alarm and event-lists, and control logic. It also includes maintenance log and plans, software publisher services for the asset firmware and related services, graphical representation of asset as used in a control room, CAD drawings, asset vendor services, etc. At the high level it gathers analytics using data related to the asset to perform long term resource planning or process optimization. Therefore we can conclude that the whole IIoT system can be subsequently seen as a system of such systems.

In Figure 8.2, a simplified generic IIoT architecture is presented, the architecture is inspired by the one described by Schriegel et. al [20]. The architecture is based on the automation pyramid, extended with some of the concepts from a typical IIoT system. As can be seen from this simplified architecture, many of the functions traditionally kept in the IT network now can be realized in an on-site or remote cloud, with applications like Condition Based Monitoring (CBM), Product Life-cycle Management (PLM) and Manufacturing Execution Systems (MES). Data can be flowing directly from devices to cloud, or via edge nodes, allowing shorter analysis/decision cycles. There might be third party vendors collecting and possibly sharing information on assets (via Vendor Cloud). Different services/devices might be implemented with different cloud architectures in mind, requiring cross cloud integration (i.e., Ext. Data cloud). There might be local or remote software publishing services for patch management, adding functionality or enabling interoperability (i.e., App Store in the cloud). There might even be control logic being executed in a cloud or edge-device. Many of the characteristics of the traditional automation pyramid do no longer hold such as:

1. No strict and predefined communication paths following the hierarchical levels.

2. There might be real-time requirements at many levels.

3. Possible mix of OT and IT functionalities at any level.

Based on these assumptions in an IIoT architecture we depict in Figure 8.2, we take a look at the standard and discuss the parts of the standard mostly impacted by this change.

8.4.2 Security zones and network segmentation in IIoT

The concept of security zones is central in IEC 62443. Given a heterogeneous IIoT system containing numerous interconnected devices and services that also utilize cloud technologies, one can raise the question whether the idea of zoning is still valid. Considering the brownfield scenario, where devices or services are introduced into one security zone, and those devices have network connectivity to other less protected zones, that will at least make the zone more susceptible to attacks. However, if components used for controlling a critical process are still isolated in a separate zone, and IIoT devices or services used for monitoring the critical process are kept in another network, the dividing into security zones clearly provide additional safety. In the Reference Architecture for Industry 4.0 (RAMI4.0) [6], it is suggested that there should be separate networks for direct process control.

In IEC 62443-3-1, the guidance (c. 6.2.7) states that only network traffic directed from the IACS towards the IT-network should be allowed. To make use of many of the advantages promised by IIoT, analytics will in many cases be performed in e.g., a cloud environment. Results from the analysis could be an updated configuration for a device to trim performance, including altering set-points. For this to work through such a firewall the communication protocol would need access to the device itself or a related service to regularly request the analytic engine for e.g., updated configurations.

Keeping network segmentation rules intact can be a challenge considering an increasing amount of the devices in the control system being IIoT devices with services distributed over the device-to-cloud continuum. Considering SR 5.2 - Zone boundary protection stating (at SL 2), network traffic crossing a zone boundary should be denied by default and allowed by exception only. Implementation of this will require a considerable amount of configuration efforts for every IIoT device added to the control system.

SR1.13 in IEC 62443-3-3 discusses access via an untrusted network, requesting the control system to monitor and control all access via such a network. In principle the guidance is that such communication paths should not exist, and if they exist, the control system should have capabilities to disable them. Both wireless and possibly untrusted networks will be a common interaction point for IIoT devices. SR1.6 and SR1.13 will in many ways be contradictory to allow some of the basic functionalities of an IIoT. These requirements could possibly be adapted so that communication over untrusted networks could be allowed, if the devices themselves fulfill specific requirements.

A novel network technology for an IIoT system with increasing popularity is Software Defined Networks (SDN), discussed in [21, 20, 1]. SDN is adopted from cloud computing technologies, and is characterized by dynamic configuration of the network by a central node, with the aim to optimize performance based on current application. This approach fits quite well with the dynamic nature of interconnections between devices and services in Industry 4.0, where applications may shift and communication paths may not be well known in advance. However, this technology seems to be in conflict with the physically or logically well defined and separated networks being protected by physical firewalls in strategic nodes as prescribed by the IEC 62443 standard.

Considering the IIoT paradigm where the communication paths are not confined within isolated networks, the need to use end-to-end security is apparent [8]. There are several cryptographical methods emerging that are relatively low-cost with regards to computational and bandwidth utilization, e.g. compressed versions of DTLS [18], which could enable using end-to-end security as a standard in IACS components. For some constrained devices, end-to-end security may still prove too costly with regards to resource consumption. In such cases specific edge nodes can be used to provide security functions for a collection of constrained devices.

8.4.3 Patch management in IIoT

The patch management guidelines, described in IEC 62443-2-3, seems to be infeasible in a number of situations when used in an IIoT system:

1. The number of devices and services involved in IIoT substantially exceeds that of a typical IACS, making the work of monitoring and updating devices infeasible;
2. A fair share of the IIoT devices will be Internet-facing or at least communicate using wireless technology, meaning that a postponed or deferred security-related update for a device could lead to an unacceptable risk of the device being compromised;
3. For the devices or services not being directly involved in controlling the physical process, following these guidelines may be too strict.

Because of the high cost and effort compared to the risk of not applying a specific patch, decisions often weighs in favour of not applying the patch, or at least delaying it until a planned maintenance stop. As a consequence, many executing IACS are not being patched to the most recent software versions, both with regards to OS and application software, potentially resulting in:

- Decreasing $SL(achieved)$ that increases the risk of the IACS being compromised;
- Incompatibilities between system parts;
- Degradation of system performance and reliability.

Secure patch management is of increasing importance in the IIoT system, but the suggested guidelines are both too strict in some sense and not strict enough in other. For an IIoT systems, there might be a need to classify devices and services based on criticality, and for the less critical components to allow, or even require, automatic patch management, e.g., based on TUF [3] or similar methodology that ensures update integrity. There are new guidelines, methods, and protocols being developed that address secure patch management. For example, the IETF Secure Update of IoT-devices (suit) work group is currently working on an architecture related to this [14].

There is an ongoing trend in software development towards DevOps [5], that most likely will affect the release cycles of some components in an IACS. DevOps is a result of combining agile software development methods with IT operations, shortening the development life-cylce and thereby the releases of a component will be more frequent and possibly without any specific periodicity. Typically a published code will push for an automatic build after which automatic tests are executed and the software is packaged. If test results are acceptable the update can be released, and possibly automatically pulled by the device instances running the software.

Another trend in software development gaining in interest in the last five years, that might impact how patch management will work in the future IACS, is the shift from classical virtualization using a hypervisor towards containerized services. Since a container execution environments provides some of the benefits from virtualization, without bringing in the overhead of emulating the OS, it could be useful as providing service execution at simpler host devices [19], e.g., the ABB Ability Edge relies on the Docker container environment.

Both DevOps and a container technology will push towards automatic patch management. For an IACS owner this will lead to increased simplicity for the technical work related to patch management, but will add a risk of less control over the system. Future version of IEC 62443-2-3 could include guidelines on how to maintain and monitor a system comprised of heterogeneous devices and services, as well as include a description on requirements for an automatic secure patch management method. Facilitating automated patch management

could help in preserving the achieved security level for the system, as well as decreasing the amount of time a known bug prevails in a specific component.

8.4.4 System/component requirements and security technologies in IIoT

When assessing the requirements in detail, the majority remains applicable in an IIoT perspective, as well. Some might however need to be revised within the new context.

The standard only briefly mentions the need for service authorization, stating that this is usually not implemented and/or used in IACS. For IIoT, this will be of great importance, as most of the interactions will be machine-to-machine.

Host firewalls are also discussed as being not commonly used in the IACS environment, as IACS product vendor typically do not allow it, along with any other third party SW, since it might affect the operability of the IACS. In IIoT systems, it would be natural at least to require that devices with direct Internet connectivity deploy micro-firewalls for added protection. Intrusion detection and prevention systems could also form an important line of defense, however, for these systems to work effectively in an IIoT environment, the cost must be lowered and the monitoring must be highly automated. The IDS and firewalls will also face an increasing amount of encrypted traffic, making statefull packet inspection more difficult when employed at intermediate network nodes, possibly deterring their effectiveness in e.g., attack-pattern recognition.

In the perspective of IIoT, both symmetric and public key encryption will be needed for some of the data-flows, especially for sensitive information that must be transferred to cloud storage for e.g., Big Data analysis. However, in traditional IACS, encryption is rarely used. Using encryption mechanisms comes with a cost both on bandwidth and CPU utilization - especially with regards to asymmetric cryptography. It is therefore of importance to assess the required protection level for specific sets of data, so that the appropriate algorithm is chosen. In the guidance from IEC 62443-3-1 with regards to encryption, it is acknowledged that encryption technologies will be of growing importance in the future, increasingly connected IACS, but the guidance currently only covers symmetric cryptography. It is suggested that any devices utilizing cryptography should be certified according to some well known security standard, e.g., FIPS 140-2 [17], to provide probability that the cryptographic algorithms used are implemented according to the state of the art. This may be a good guidance, but it will possibly prove difficult to follow for any device and service developer. It could be argued that compliance to

SDLA, or evidence of using an industry standard approved cryptographic libraries can be used to strengthen trustworthiness of IACS components using cryptography.

Audit logs for user activities are discussed, e.g., access control events, however, for devices or services activities, audit logs are not discussed in depth, but should be of increasing importance from an IIoT perspective. Especially regarding automatic collection and analysis of audit logs combined with an automatic counter-act system for detected anomalies. This should be useful in a scenario with a large number of access points. Exactly what information should be logged regarding machine-to-machine communication could be elaborated. The guidance states that security related data e.g., user account creation or failed logins should be logged, but for the IIoT scenario there might be additional information that are of interest, e.g., device discovery and disconnect, protocol handshakes resulting in a protocol version degradation, etc.

A vulnerability scanning for IIoT-devices could be a useful way of assessing the device security characteristics. To enable a vulnerability scanner in an IIoT system, the information needed to understand how to perform a scan and classify vulnerabilities with regards to a wide range of devices with widely different execution environments should exist. In the guidance, vulnerability scanners are suggested to be used mainly on hosts running standard operating systems.

Host Configuration Management (HCM) tools for centrally managing resources and user accounts are discussed in the standard as not being widely used in IACS. Due to the heterogeneous nature of an IACS system, current HCM tools are not adequate as they typically target only one kind of operating system. For an IIoT system, the diversity of devices will be even a bigger issue, at the same time as the need for efficient and centralized management is of great importance. Possibly parts of this management will be automatically executed in an IIoT system.

8.5 Conclusions

IEC 62443 is a well known and widely used standard within industrial automation. It describes requirements and the best practice for development, integration and assessments of components and systems related to an IACS with regards to cybersecurity. The emergence of the IIoT paradigm adds a new dimension to be considered compared to traditional IACS. Given the

expected complexity of such systems, our aim was to perform an analysis of the IEC 62443 standards and asses its applicability with regards to IIoT. We have noticed that several parts of IEC 62443 are already well suited for use in the context of IIoT systems. However, a number of concepts as described in the standard may prove difficult to comply with, specifically including

1. Security zone boundaries will be more difficult to withhold due to the dynamic characteristics of an IIoT system.
2. Communication over zone boundaries will be a requirement for many IIoT devices and services in order to provide any value, something which is currently discouraged by the standard.
3. For software updates, a significant level of automation of updates will be needed for IIoT devices and services. It is currently not described in the standard if and how such automation can be supported.

Apart from additional guidance on these challenges, there is also a need for technology that might not yet be available, e.g., micro-firewalls for IIoT devices, vulnerability scanners for IIoT systems, HCM tools spanning IIoT devices and services.

As future work, it may additionally be useful to take a look at related standards and recommendations, compare and potentially get inspiration for complementing IEC 62443. Examples of relevant related standards include the NIST Framework for Improving Critical Infrastructure Cybersecurity [16] and the suit architecture for secure updates of IoT devices [14]. Additionally, Industrial Internet Consortium has developed a security framework as a part of its reference architecture (IIC IIRA G4) [8].

Acknowledgements

This work is supported by the industrial postgraduate school Automation Region Research Academy (ARRAY), and the SAFSEC-CPS project, both funded by The Knowledge Foundation. In addition, ABB has provided funding. The authors would like to acknowledge Judith Rossebø and Tomas Lindström for their feedback.

Bibliography

[1] S. Al-Rubaye, E. Kadhum, Q. Ni, and A. Anpalagan. Industrial internet of things driven by sdn platform for smart grid resiliency. *IEEE Internet*

of Things Journal, 6(1):267–277, Feb 2019.

[2] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson. The industrial internet of things (IIoT): An analysis framework. *Computers in Industry*, 101:1–12, June 2018.

[3] Cappos, Justin et. al. The Update Framework. `https://theupdateframework.github.io/`, 2019. [Online; accessed May 13, 2019].

[4] CMMI Institute. CMMI services. `https://cmmiinstitute.com/cmmi/svc`, 2019. [Online; accessed May 14, 2019].

[5] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano. Devops. *IEEE Software*, 33(3):94–100, May 2016.

[6] IEC. Smart Manufacturing - Reference Architecture Module Industry 4.0 (RAMI4.0). Technical report, Internation Electrotechnical Commission, 2016.

[7] IEC 62443 security for industrial automation and control systems. Standard, Internation Electrotechnical Commission, Geneva, CH, 2009-2018.

[8] IIC. The Industrial Internet of Things Volume G4 : Security Framework. Technical report, Industrial Internet Consortium, 2016.

[9] ISA Security Compliance Institute. Security Development Life-cycle Assurance - Certification Requirement, rev 1.3. Technical Report SDLA-300, june 2014.

[10] IEC 62443-4-2 - EDSA Certification. `http://www.isasecure.org/en-US/Certification/IEC-62443-Edsa-Certification`, 2019. [Online; accessed May 2, 2019].

[11] R. Kissel. *Glossary of key information security terms, Revision 2.* U.S. Dept. of Commerce, National Institute of Standards and Technology, 2013.

[12] W. Knowles, D. Prince, D. Hutchison, J. F. P. Disso, and K. Jones. A survey of cyber security management in industrial control systems. *International Journal of Critical Infrastructure Protection*, 9:52 – 80, 2015.

[13] R. M. Lee, M. J. Assante, and T. Conway. Analysis of the Cyber Attack on the Ukrainian Power Grid. Technical report, SANS, 2016.

[14] B. Moran, M. Meriac, H. Tschofenig, and D. Brown. A Firmware Update Architecture for Internet of Things Devices. Internet-Draft draft-ietf-suit-

architecture-05, Internet Engineering Task Force, Apr. 2019. Work in Progress.

[15] NERC. NERC CIP Standards. `http://www.nerc.com/pa/Stand/pages/cipstandards.aspx`, 2019. [Online; accessed May 9, 2019].

[16] NIST. Framework for Improving Critical Infrastructure Cybersecurity. Technical report, 2018.

[17] NIST. Security Requirements for Cryptographic Modules. Technical report, 2019.

[18] S. Raza, H. Shafagh, K. Hewage, R. Hummen, and T. Voigt. Lithe: Lightweight secure coap for the internet of things. *IEEE Sensors Journal*, 13(10):3711–3720, Oct 2013.

[19] J. Rufino, M. Alam, J. Ferreira, A. Rehman, and K. F. Tsang. Orchestration of containerized microservices for IIoT using Docker. In *Proceedings of the IEEE International Conference on Industrial Technology*, pages 1532–1536. IEEE, 2017.

[20] S. Schriegel, T. Kobzan, and J. Jasperneite. Investigation on a distributed SDN control plane architecture for heterogeneous time sensitive networks. *IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS*, 2018-June:1–10, 2018.

[21] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. V. Vasilakos. Software-defined industrial internet of things in the context of industry 4.0. *IEEE Sensors Journal*, 16(20):7373–7380, Oct 2016.

[22] T. J. Williams. The purdue enterprise reference architecture. *Computers in Industry*, 24(2):141 – 158, 1994.

Article C

Chapter 9

Article C: Access Control for Smart Manufacturing Systems

Björn Leander, Aida Čaušević, Hans Hansson, Tomas Lindström
In the proceedings of the 14th European Conference on Software Architecture, 2nd Workshop on Systems, Architectures, and Solutions for Industry 4.0 (SASI4), L'Aquila, Italy, September 2020

Abstract

In the ongoing 4^{th} industrial revolution, a new paradigm of modular and flexible manufacturing factories powered by IoT devices, cloud computing, big data analytics and artificial intelligence is emerging. It promises increased cost efficiency, reduced time-to-market and extreme customization. However, there is a risk that technical assets within such systems will be targeted by cybersecurity attacks. A compromised device in a smart manufacturing system could cause a significant damage, not only economically for the factory owner, but also physically on humans, machinery and the environment.

Strict and granular Access Control is one of the main protective mechanisms against compromised devices in any system. In this paper we discuss the requirements and implications of Access Control within the context of Smart Manufacturing. The contributions of this paper are twofold: first we derive requirements on an Access Control Model in the context of smart manufacturing, and then asses the Attribute Based Access Control model against these requirements in the context of a use case scenario.

9.1 Introduction

Smart manufacturing [14, 2] is a development of traditional manufacturing implying a shift from production of big series of identical units towards a highly dynamic manufacturing environment that is tuned to extreme customization, fluctuating markets, and specific customer needs. The technology to enable this dynamic behavior includes an increasing amount of interconnected sensors, actuators and related services in the manufacturing environment, in combination with e.g., cloud technologies, data lakes, artificial intelligence, etc., for inference and aid to decision-makers [22].

In the dynamic smart manufacturing environments of today and tomorrow, the traditional view of the manufacturing networks being air-gapped and protected by proprietary technologies no longer holds [23]. Considering that a great number of these devices introduced in a smart manufacturing system have wireless connectivity, are living on the edge of the network, possibly with direct connections to unprotected networks, there is an increasing risk that any of these devices become compromised. This has been illustrated in a number of attacks targeting industrial systems over the last ten years [21]. To protect the manufacturing environment from compromised devices, there is a need to introduce a number of security measures in the form of e.g., Intrusion Detection Systems (IDSs), end-to-end security for sensitive data, malware detection and fine-grained access control.

In this article we focus on *access control*, as one of the basic security functions in any system, enabling access restriction to operations on resources only to legitimate authorized subjects. The models for access control that are currently in use are tailored to authorize human subjects performing operations on digital assets, mainly supporting use-cases for rather static sets of resources and subjects or roles. These traditional models do not provide a high level of flexibility for expressing fine-grained policies [25], as frequently needed in smart manufacturing. Attribute Based Access Control (ABAC) is a relatively new model for policy formulation, potentially useful for machine-to-machine authorization [15, 10]. Our aim in this paper is to derive requirements on access control in smart manufacturing systems, and evaluate ABAC against those requirements.

The remainder of this paper is structured as follows: Background is presented in Section 9.2. In Section 9.3 we identify a compilation of requirements on access control. In Section 9.4 a use cases scenario for smart manufacturing is presented, including suggestions on policy formulations for ABAC in this context. A discussion on how ABAC relates to the requirements are provided in

Section 9.5. Scientific work related to our findings is presented in Section 9.6. Finally the work is summarized and some remaining challenges and future areas of research are described in Section 9.7.

9.2 Background

9.2.1 Smart Manufacturing concepts

The term *smart manufacturing* is used for describing the 4^{th} industrial revolution from a manufacturing perspective, with origin in a joint work by several agencies in the US [22]. Smart manufacturing is sometimes also referred to as Cyber Physical Production Systems (CPPS) [16] and Intelligent Manufacturing Systems (IMS)[1].

In general, smart manufacturing encompasses the whole manufacturing chain, from supply to production and logistics. Data collected from sensors within the process are used for advanced data analytic in order to improve the overall operations. A key aspect of smart manufacturing is to provide flexibility and dynamicity in the manufacturing environment by modularization of process steps, so that process steps can be combined and re-combined based on current production requirements [13]. Integrating modular process steps in the manufacturing system enables Workflow as a Service (WfaaS), where vendors of production equipment could sell pre-fabricated process-steps as a service, allowing the factory owners to easily adapt to fluctuating market demands.

9.2.2 Cybersecurity Threats to Smart Manufacturing Systems

The increasing amount of connected and interconnected devices required for the data acquisition together with external stakeholders in need to access the data, considerably increases the attack surfaces of a smart manufacturing system. Furthermore, as different modules within the system are dynamically connected to each other, the authorization of privileges between devices and services must be equally dynamic to allow continuous secure operation. According to Tuptuk et al. [23], cybersecurity is rather seen as a characteristic than as a design principle within the development of smart manufacturing systems, a misconception that may lead towards many systems being insufficiently protected.

The CIA-model is often used to describe desired security characteristics of

[1]More information available at `http://ims.org`.

a system (**C**onfidentiality, **I**ntegrity and **A**vailability [26]). In the context of smart manufacturing, a cybersecurity attack may breach any of these characteristics, e.g., leading to possible loss of Intellectual Property (IP), costly errors in production due to unreliable or faulty data, and down-time or potentially safety-related threats to production machinery, workers amd the environment.

9.2.3 Access Control definitions

There are a number of guiding principles for access control, the most notable ones being [18]:

1. **Least privilege**, requires that a subject should only have the minimum possible privileges needed to perform its tasks.
2. **Complete mediation** requires that any access to a resource must be monitored and verified.

Following these principles in a smart manufacturing system will help minimize the harm an adversary can do after gaining an initial foothold within the system, and even shorten the detection time, since failed access attempts typically are logged and monitored.

Sandhu el al. [19] describe access control as being comprised of models at three different layers, **P**olicy, **E**nforcement and **I**mplementation (PEI). Policy models are used to formalize high level access control requirements, enforcement level models describe how to enforce these policies from a systems perspective, and the implementation level models show how to implement the components and protocols described by the enforcement model. Following the PEI-model, this work is focusing on the policy-layer models, meaning that we will discuss how rules can be expressed, rather than mechanisms to enforce the rules.

A prerequisite for robust access control is reliable authentication of entities. In this work we assume that a trustworthy solution for authentication is used.

Historically, Mandatory Access Control (MAC) and Discretionary Access Control (DAC) have been the two main paradigms within access control [20]. MAC is based on security classifications of resources, combined with security clearances for subjects, e.g., top-secret content only readable for subjects with the highest security clearance. In DAC on the other hand, the privileges are defined as a relation between the resource and subject, often with the subject allowed to transfer its privileges.

Role-Based Access Control (RBAC) is a model building on principles from both DAC and MAC, where subjects are assigned to one or several roles that may be hierarchically ordered. Privileges are derived from the roles rather than from the subject. In a number of studies it has been shown that the traditional access control schemes are not sufficient for, e.g., cloud-connected cyber physical systems [12] and IIoT [17].

9.2.4 Attribute Based Access Control (ABAC)

A relatively novel scheme in access control is ABAC. In the work of Yuan and Tong [29], the application is aimed at providing access control in web services. They show that the granularity of the traditional RBAC scheme is not fine enough, in order to formulate certain policies easily expressed in natural language. The following example is extracted from [29], and provided here to introduce ABAC and illustrate that such natural language rules are difficult to express using the traditional Access Control models:

Let us assume we need to grant a user access to movies in an online streaming service. In this example we consider a movie $rating$ (R, R-13, G) and $freshness$ (New release, Normal), mapped to the user age and subscription $category$ (Budget, Premium). The following to rules apply for a user to be allowed to watch a movie:

1. To watch movies with rating R, user must be over 17 years old, and for movies with rating R-13, over 12 years.
2. To watch a New release, the user subscription category must be Premium.

In ABAC, the subject s's right to perform operation o on a resource r in environment e is calculated based on attributes of the subject, resource and environment, A_s, A_r, A_e respectively:

$$allow_o(s, r, e) \leftarrow f(A_s, A_r, A_e)$$

For the movie streaming service, the following policy rules can be expressed, based on the viewer and movie attributes:

$$f_1(s, r, e) = \Big(rating(r) = \text{G}\Big) \vee \Big(age(s) > 12 \wedge rating(r) = \text{R-13}\Big) \vee \Big(age(s) > 17\Big) \quad (9.1)$$

$$f_2(s, r, e) = \Big(freshness(r) = \text{normal}\Big) \vee \Big(category(s) = \text{premium}\Big)$$

allowing for rules to be further combined:

$$allow_{view}(s, r, e) = f_1(s, r, e) \wedge f_2(s, r, e).$$

Several works on ABAC have been conducted, including two major standardization efforts in the area: eXtensible Access Control Markup Language (XACML) by OASIS [28], and Next Generation Access Control (NGAC) by NIST [5]. A comparison between NGAC and XACML is provided by Ferraiollo et al. [4].

Authorization architectures for ABAC typically contain a number of standard components [10, 4, 28]: A subject can only access a resource through the the Policy Enforcement Point (PEP), which acts as a mediator for any privilege request. The PEP queries an authorization decision from the Policy Decision Point (PDP) that reads policy information from the Policy Information Point (PIP), which has access to Policy Data. An administrator maintains Policy Data through a Policy Administration Point (PAP).

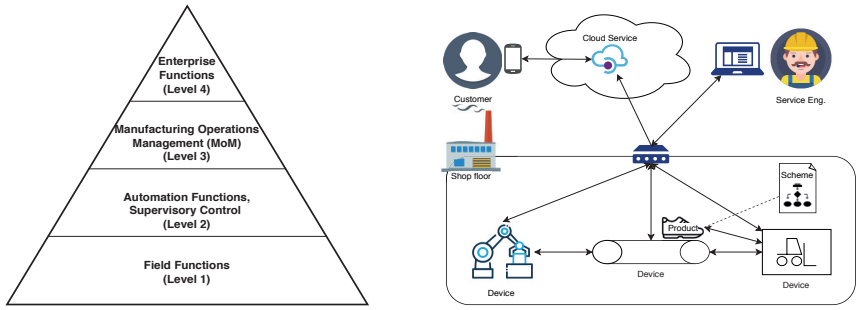
9.3 Access Control Requirements on Smart Manufacturing

In this section we formulate a list of requirements on access control for a smart manufacturing system. To provide such a list we have studied the literature, using an adapted version of the method presented by Kitchenham [8]. We have selected relevant requirements guided by the basic principles for access control. For details regarding the literature review and used protocol we refer the reader to [11].

9.3.1 Requirements related to a traditional manufacturing system

A traditional manufacturing system can be described as an Industrial Automation and Control System (IACS) which typically supports safety- and security critical processes [7]. IACS are used to control and monitor a wide range of different types of physical processes, e.g., in chemical industries, power plants, and discrete manufacturing.

An illustration of a generic traditional manufacturing system architecture can be seen in Figure 9.1a, inspired by the Purdue Enterprise Reference Architecture (PERA) [27]. These systems contain a number of *essential functions* that cannot be disrupted, and that are required to maintain health, safety and



(a) A traditional manufacturing system architecture based on PERA

(b) A simplistic smart manufacturing scenario

Figure 9.1: An overview of a traditional and a smart manufacturing architecture.

availability of the equipment under control. In principle, a security measure must not result in a state of the system that could lead to Health, Safety or Environmental (HSE) consequences. A number of requirements on the access control arise from the need to support essential functions [7]:

R1 Availability: The manufacturing system should be operable even if some components fail, e.g., a failed server or a disruption in network connectivity between shop floor and cooperate network should not interfere with production.

R2 Security measures must not have a negative impact on essential functions. Specifically, HSE-related incidents shall not happen as a result of loss of control due to lack of privileges.

Non-Repudiation is also an important characteristic of access control that is required by e.g., IEC 62443[2]. We choose not to list it as a requirement in this context, as the focus of this work is on mechanisms for access control at a policy level and non-repudiation refers to logging and auditing of execution of granted privileges.

9.3.2 Requirements related to smart manufacturing systems

A number of requirements on access control are shared between the smart manufacturing domain and other dynamic systems of interconnected cyber-physical systems. These requirements arise through the evolution of the traditional automation pyramid towards a service oriented and decentralized system [13, 12]:

[2]Part 3-3: System security requirements and security levels, Ed 1.0, 2013

R3 Diversity: A system should provide support for several different kinds of applications to be integrated throughout the whole life-cycle. This implies that multiple categories of users, usages of services and production related data shall be supported by the system.

R4 Scalability: A system should be scalable with regards to users and policies. Management of a huge amount of devices, services and users must be simple and cost efficient, still providing necessary transparency.

R5 Flexibility: The access control mechanism shall provide an easy way of defining new policies.

R6 Efficiency: The computational cost of inferring privileges should not negatively impact the performance of the system as a whole.

From [17, 9, 3, 1] we have derived the following requirements specific to the smart manufacturing domain:

R7 Temporal policies: The required privileges to perform a task may shift between each batch, or even between each produced unit. The access control model shall be equally flexible, following the principle of least privilege.

R8 Logical ordering: Production in a manufacturing environment is usually described as a workflow, meaning that the order of the actions, and the number of times an action can be executed could be limited. The access control model shall be able to express such logical ordering at a policy level.

9.3.3 Generic access control requirements

In the following we describe generic access control requirements not covered in earlier sections. These requirements are the result of discussions with industrial experts:

R9 Transparency: From the perspective of an administrator, it must be easy to deduce current state of granted privileges, and historical changes to privileges. This transparency requirement could also extend to other privileged users.

R10 Delegation: For certain scenarios, it should be possible to transfer privileges from one subject to another through delegation.

9.4 A Smart manufacturing Scenario

In this section we describe a generic smart manufacturing scenario to be analyzed from an access control perspective. We provide a discussion on how ABAC can be applied to the scenario in Figure 9.1b. The scenario essentially follows the set-up of a service-driven architecture for manufacturing, described in [13, 3], connected to the IEC 61499 [6].

Let us assume that a product p is to be manufactured. p is associated to a set of devices $\mathbf{D}$ that must perform tasks on p for it to be finalized. In order to perform the actions there is a need for a device $d \in \mathbf{D}$ to share information, and execute operations on one or more other devices in $\mathbf{D}$, according to the manufacturing scheme defined specifically for p.

The customer c wants to read information from the system for data related to product p via a cloud service, e.g., production status and expected delivery time. A 3rd party service organization o is responsible for maintaining some of the devices in $\mathbf{D}$, and must therefore be able to read status and perform service-related actions on the devices, e.g., reading health records and performing firmware upgrades.

In practice, the rules we describe in the following would be implemented using e.g., XACML [28]. For brevity, we choose to describe only the logical expressions of the policies, following the formalism introduced in [29]. The following attributes will be used in the ABAC policy formulations below:

- $batch_{id}(x)$[3] is the value of the batch attribute, related to a produced entity p or related to the current context of execution for a device d.
- $batches(e)$ is the set of all active batches in the manufacturing environment e.
- $purchases(c)$ is the set of batches that customer c has purchased. In this example we assume a one-to-one connection between customer and batch.
- $contract_{id}(d)$ is the value of the service-contract attribute related to a device d.
- $contracts(o)$ is a set of contracts under which the service organization o is working.

[3]Here x is used as variable representing either an entity p or a device d.

- $idle(d)$ is a Boolean attribute indicating that device d is currently idle if TRUE or busy if FALSE.
- $*$ is used to indicate an unassigned attribute value.

Given the example, we are able to show some interesting characteristics regarding access control in smart manufacturing systems.

C1 Machine to Machine (m2m) cooperation is limited by the current entity/batch attribute.

C2 Customer outside organization read rights are limited by a purchase.

C3 Service organization personnel (possibly 3rd party) having read and e.g., firmware-update rights limited by a contract.

Using ABAC, a policy to satisfy characteristic C1 could be expressed as:

$$allow_{op}(d_1, d_2, e) = \Big(batch_{id}(d_1) = batch_{id}(d_2)\Big) \wedge batch_{id}(d_1) \in batches(e) \quad (9.2)$$

Stating that the privilege to perform the operation will be granted only if the devices d_1 and d_2 have the attribute $batch_{id}$ assigned with the same id, and that id is among the active batches in the environment. Similarly, the customer could be granted privileges based on a combination of attributes of the data and attributes of the customer, which would allow a very fine-grained model for authorization (i.e., related to characteristic C2). One simple example of an authorization rule could be:

$$allow_{read}(c, p, e) = batch_{id}(p) \in purchases(c) \quad (9.3)$$

Note that in this specific rule, as well as the following, no environment attributes are used. Entity e will still be used in the declaration of the formula for consistency reasons. The above equation is stating that reading information about product p is allowed if the $batch_{id}$ for p is present in the set of $purchases$ that the customer c has done. Typically such information is retrieved through filtering, i.e., the privilege is enforced by the application or API implementation, which is a much weaker condition than granting privileges through the access control mechanism. In fact, following the traditional practice, the access control mechanism will grant read-access to any valid customer and rely on the application to perform the correct filtering.

The privileges of personnel from the service organization (i.e., related to characteristics C3) is an interesting issue, since there may be many factors within

the manufacturing environment that should prevent interruption or additional load on devices or services related to direct operation. In a classical service operation scheme, privileges to perform maintenance related operations may not be allowed except when the production unit is halted for planned maintenance or similar. However, in a smart manufacturing environment, this may be a common case, especially for WfaaS scenarios, i.e., it is up to the service organization to make sure that the workflows are running as needed. In these cases, an ABAC policy could be used to minimize the risk of disturbing ongoing operations. For example, an attribute indicating that the device is currently in use could inhibit the right to perform disruptive actions, and attributes indicating a need to perform an update or a similar disruptive maintenance action could inhibit the device from being assigned to a batch. The following rule could be set up for intrusive service operations:

$$allow_{op}(o, d, e) = \Big(contract_{id}(d) \in contracts(o)\Big) \wedge idle(d) \tag{9.4}$$

Stating that the operation is allowed if the service contract for the device d is in the set of contracts the service organization o is working under, and d is currently idle.

9.5 Fulfillment of requirements

A summary of the requirements and the fulfillment levels with regards to ABAC is provided in Table 9.1. The fulfillment level *Fulfilled* denotes that ABAC is well suited to fulfill the requirement; *Possible* denotes that fulfillment is possible, but depends on the implementation; and *Unclear* denotes a requirement where the fulfillment level is difficult to assess from available documentation. In the following we discuss the reasoning behind the fulfillment assessment.

ID	Requirement	Description	Fulfillment
R1	Availability	Work in spite of degraded functionality	Possible
R2	Critical Events	No HSE impact	Possible
R3	Diverse	Many user categories and usages of services and data	Fulfilled
R4	Scalable	Management of huge amount of devices, services, users	Unclear
R5	Flexible	AC must allow easy policy creation for new scenarios	Fulfilled
R6	Efficient	Cost of AC cannot impact system performance	Unclear
R7	Temporal policies	Quick shift in policies, due to customization	Fulfilled
R8	Logical Ordering	Workflow based access control	Unclear
R9	Transparency	Administrator to see what privileges are granted and why	Possible
R10	Delegation	Privileges transferable through delegation	Possible

Table 9.1: Requirements fulfillment for ABAC

ABAC is able to express fine-grained rules due to the use of attributes on subjects, objects and the environment, as well as the possibility to set up policy-rules as functions of these attributes. This granularity and expressiveness will allow a very high level of flexibility, leading to fulfilling **requirement R5**. As illustrated in the Section 9.4, it seems possible to express rules in ABAC so that the principle of *least privilege* is satisfied, something that would be more challenging using e.g., RBAC. The **requirement R3** on diversity is also fulfilled, provided that policies can be easily added and adapted for different applications and user categories. Here the enforcement and implementation considerations are of great importance.

The reasoning used for **R5** is also valid for **requirement R7**, as it arises as a result of quick shifts in policies, due to e.g. customization. Hence, it can be fulfilled since it is possible to express very fine-grained rules based on attributes. As demonstrated in the scenario description, it is possible to express policies so that they are meaningful in the context of shifting production schemes.

The management effort of an ABAC-model may not scale well with increasing size and complexity of the system (**requirement R4**). It may be the case that policy rules can be expressed in such a general way, as suggested in Section 9.4, but there are certainly more complex scenarios including a potentially larger set of rules. Any privilege request needs to evaluate all rules applicable for that specific request, demanding logic for handling combinations of rules. In a system with a complex set of policies, the implications of adding or altering a policy can be difficult to foresee. Attribute provisioning is also a management issue in a dynamic system. There is a need for trusted Attribute Authorities to provide the integrity of claimed attributes.

A low computational cost (**requirement R6**) is not a general property of ABAC. Depending on the implementation and how the policy base is formulated, the operation of granting or denying a privilege request may be computationally expensive. In case of using e.g., XCAML [28] for policy expression, there does not seem to be a bounded cost for inference [4, 24]. The total cost of inference must also include the time for attribute enumeration, which may need additional communication rounds with Attribute Authorities.

Requirement R1 implies that there should be a distributed architecture for access control in smart manufacturing applications, possibly including redundancy for key entities. This characteristic is uncommon in most available access control enforcement models. An ABAC architecture consist of several authorities, which all must be available to provide continuous privilege en-

forcement. However, it is possible to fulfill the requirement of a functioning access control mechanism during degraded mode using an enforcement architecture with local caches for attributes and policies that can be used in isolation. Another possibility is using a distributed architecture of policy- and attribute-authorities.

Requirement R2 concerns the possibility for a system to stop (e.g., operator lock-out during a critical scenario), and could possibly be met by ABAC using an environment attribute indicating a *system state* within the plant. This would however not be the first option for designing the system to protect it from HSE-related incidents. Instead, secondary control-units are typically used for essential functions, e.g., an emergency stop. Those controls are not dependent on standard user authentication and authorization, and will have a very limited functionality. Therefore, the fulfillment of this requirement can be seen as *possible*, even though it is not directly dependent on the access control model.

Requirement R8 is stating that the access policies should follow the workflow in the process. This is currently not supported by ABAC. There are mechanisms in e.g., NGAC and UCON [15] called obligations, which may alter privileges based on previous policy decisions. However, it is not clear if obligations can be used to describe a state-machine altering attribute assignments to mimic a process workflow.

A generic requirement on an access control model is to provide transparency (**requirement R9**). For ABAC it is unclear if such functionality is available neither with regards to an administrator, nor to a user. A solution on the implementation-model level could possibly be able to answer to the transparency needs of an administrator, but it is not intrinsic to the access control scheme, as in the case with e.g., the access control list (ACL) ability to perform per-resource review, or the RBAC ability to perform a per-subject review.

To be able to transfer privileges between subjects, as stipulated by **requirement R10**, is common during delegation in industrial systems. In the case of ABAC, this would require a subject to be able to transfer a set of attributes to another entity, as the privilege inference is based on attribute values. In principle there is nothing in ABAC that specifically prohibits this. However, it may prove a challenge in practice, as the subject needs to know precisely which attributes to transfer in order to achieve the intended privilege delegation. Detailed knowledge on how the policy-rules are expressed is needed to perform privilege delegation in ABAC. Looking at the examples from our use case scenario in Section 9.4, it would be quite easy to allow delegation by

e.g., transfer the $contract_{id}$ attribute to a service engineer temporarily working with maintenance under a specific contract, but there are more complex scenarios in which several rules concurrently may influence a privilege decision. Furthermore, when transferring attributes there is a need to limit the usage of the attributes to the actual scope of the delegation, otherwise there is an apparent risk that the attribute transfer will grant other privileges than was intended. Our conclusion is that additional mechanisms in the enforcement and implementation layers are needed to make this requirement practically achievable.

9.6 Related Work

Salonikas et al. discuss the concept of access control requirements in a dynamic industrial system with focus on the wider concept of IIoT [17], while Lopez et al. target cloud connected cyber physical systems [12]. Both articles discuss different access control models at the policy level, very similar to our work. However, these articles do not consider modular system characteristics specific for a smart manufacturing, as we do.

Watson et al. [25], discuss the use of different access control models in conjunction with OPC UA. They advocate the use of ABAC or a combination of ABAC and RBAC as a good match for protection against privilege escalation for both inside and outside attackers within IACS. Their work can be seen as a suggestion for the enforcement layer, whereas our work provides guidance applicable to the policy layer.

Some of the existing work present variations of ABAC suitable in different domains. Lang et al. [10] suggest a proximity based access control (PBAC), well suited for intelligent transportation systems. It originates from the ABAC model, but uses the mathematical proximity between subject and resource as one of the deciding factors for granting privileges. To derive policy rules, Model Driven Security (MDS) is used. MDS usually relies on a modeling tool in which the policy can be described at a high level of abstraction and the actual enforcement rules are then generated based on that model. Park and Sandhu [15] present the Usage CONtrol (UCON$_{ABC}$)-model, which can also be seen as an extension of the ABAC model with obligations. In this approach, an access-control event could alter attributes or conditions for future access controls. This mutability of attributes, or a variation thereof, could possibly be used to model the behavior of temporal workflows required by smart manufacturing. Next Generation Access Control (NGAC) [4] is the NIST proposal on how ABAC should be described. Compared to the tra-

ditional ABAC, in this variant attributes are provided as hierarchical labels (i.e., similar to RBAC group hierarchies), rather than properties with values as described in the initial ABAC-models. All of these approaches have interesting features useful in a smart manufacturing system, e.g., the model driven approach from PBAC and the obligations from UCON$_{ABC}$. As a future work, we aim to investigate the possibility to combine the beneficial concepts from these approaches in a practical smart manufacturing scenario.

9.7 Conclusions

Smart manufacturing is a technology that has a huge economical potential, transforming manufacturing towards servitization and extreme customization. However, the technologies that such systems are built upon bring new challenges, especially as the increasing attack surface expose the system to additional cybersecurity threats. As we have argued in this paper, one of the largely neglected mechanisms for security within manufacturing systems is access control between devices and services. Since the dynamic properties of smart manufacturing require a similarly dynamic model for access control, additional attention must be directed to this issue.

In this article we have derived a number of requirements on access control within smart manufacturing systems, based on knowledge related to traditional manufacturing systems, interconnected cyber-physical systems, and industrial expertise. These requirements are considering both the guiding principles for access control and the basic safety principles of an industrial control system.

Illustrated by a use-case scenario we have mapped the requirements to the ABAC model, and shown that the model aligns well with the requirements. However, there are still several open questions to be answered. How to handle scalability with regards to management of policies and attributes in large systems seems to be the most difficult issue to deal with, especially for complex sets of access control policies. The management process must be sufficiently light-weight in order for the model to be adopted in real applications. Transparency and efficiency are other areas where additional efforts are needed to make the ABAC model a feasible alternative for modern industrial manufacturing systems.

As future research we envision conducting a simulation study with use-cases from the smart manufacturing domain, together with e.g., the Policy Ma-

chine, which is the reference implementation of NGAC from NIST[4]. The management issue of security policy generation could possibly be handled using model driven security, as discussed by Lang et al. [10].

Acknowledgements

This work is supported by the industrial postgraduate school Automation Region Research Academy (ARRAY), funded by The Knowledge Foundation. The authors would like to acknowledge Andrea Macauda and Axel Haller for valuable discussions and feedback.

Bibliography

[1] I. Ayatollahi, J. Brier, B. Mörzinger, M. Heger, and F. Bleicher. SOA on smart manufacturing utilities for identification, data access and control. *Procedia CIRP*, 67:162–166, 2018.

[2] J. Davis, T. Edgar, J. Porter, J. Bernaden, and M. Sarli. Smart manufacturing , manufacturing intelligence and demand-dynamic performance. *Computers and Chemical Engineering*, 47:145–156, 2012.

[3] C. Faller and M. Höftmann. Service-oriented communication model for cyber-physical-production-systems. *Procedia CIRP*, 2018.

[4] D. Ferraiolo, R. Chandramouli, R. Kuhn, and V. Hu. Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC). pages 13–24, 2016.

[5] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone. Guide to Attribute Based Access Control (ABAC) Definition and Considerations. Technical report, NIST, 2014.

[6] IEC 61449 function blocks. Standard, Internation Electrotechnical Commission, Geneva, CH, 2012.

[7] IEC 62443 security for industrial automation and control systems. Standard, Internation Electrotechnical Commission, Geneva, CH, 2009-2018.

[8] B. A. Kitchenham. Procedures for Undertaking Systematic Reviews. Technical report, Keele University, 2004.

[4] `https://csrc.nist.gov/Projects/Policy-Machine`

[9] J. Ladiges, A. Fay, T. Holm, U. Hempen, L. Urbas, M. Obst, and T. Albers. Integration of modular process units into process control systems. *IEEE Transactions on Industry Applications*, 54(2):1870–1880, March 2018.

[10] U. Lang and R. Schreiner. Proximity-based access control (PBAC) using model-driven security. In H. Reimer, N. Pohlmann, and W. Schneider, editors, *ISSE 2015*, pages 157–170, Wiesbaden, 2015. Springer Fachmedien Wiesbaden.

[11] B. Leander. Towards an access control in a smart manufacturing context. Technical report, Mälardalen Real-Time Research Centre, Mälardalen University, 2020.

[12] J. Lopez and J. E. Rubio. Access control for cyber-physical systems interconnected to the cloud. *Computer Networks*, 2018.

[13] Y. Lu and F. Ju. Smart Manufacturing Systems based on Cyber-physical Manufacturing Services (CPMS). *IFAC-PapersOnLine*, 50(1):15883–15889, 2017.

[14] S. Mittal, M. A. Khan, and T. Wuest. Smart manufacturing: Characteristics and technologies. In R. Harik, L. Rivest, A. Bernard, B. Eynard, and A. Bouras, editors, *Product Lifecycle Management for Digital Transformation of Industries*, pages 539–548, Cham, 2016. Springer International Publishing.

[15] J. Park and R. Sandhu. The UCON$_{ABC}$ usage control model. *ACM Transactions on Information and System Security*, 7(1):128–174, 2004.

[16] A.-R. Sadeghi, C. Wachsmann, and M. Waidner. Security and privacy challenges in industrial internet of things. *Proceedings of the 52nd Annual Design Automation Conference on - DAC '15*, pages 1–6, 2015.

[17] S. Salonikias, A. Gouglidis, I. Mavridis, and D. Gritzalis. Access control in the industrial internet of things. In C. Alcaraz, editor, *Security and Privacy Trends in the Industrial Internet of Things*. Springer International Publishing, 2019.

[18] J. Saltzer and M. Schroeder. The Protection of Information in Computer Systems. In *proceedings of the IEEE*, volume 63, pages 1278–1308, September 1975.

[19] R. Sandhu, K. Ranganathan, and X. Zhang. Secure information sharing enabled by trusted computing and PEI models. *Proceedings of the*

2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS '06, 2006:2–12, 2006.

[20] R. S. Sandhu and P. Samarati. Access control: Principles and Practice. *IEEE Communications Magazine*, 32(September):40–48, 1994.

[21] J. Slowik. Evolution of ICS Attacks and the Prospects for Future Disruptive Events. Technical report, 2017.

[22] K.-d. Thoben, S. Wiesner, and T. Wuest. "Industrie 4.0" and Smart Manufacturing – A Review of Research Issues and Application Examples. *International Journal of Automation Technology*, (January), 2017.

[23] N. Tuptuk and S. Hailes. Security of smart manufacturing systems. *Journal of Manufacturing Systems*, 47(April):93–106, 2018.

[24] F. Turkmen and B. Crispo. Performance evaluation of XACML PDP implementations. *ACM Conference on Computer and Communications Security*, 2008.

[25] V. Watson, J. Sassmannshausen, and K. Waedt. Secure Granular Interoperability with OPC UA. In C. Draude, M. Lange, and B. Sick, editors, *INFORMATIK 2019: 50 Jahre Gesellschaft für Informatik – Informatik für Gesellschaft (Workshop-Beiträge)*, pages 309–320, Bonn, 2019. Gesellschaft für Informatik e.V.

[26] M. Whitman and H. Mattord. *Principles of Information Security*. Cengage Learning, 4th edition, 2012.

[27] T. J. Williams. The purdue enterprise reference architecture. *Computers in Industry*, 24(2):141 – 158, 1994.

[28] eXtensible Access Control Markup Language (XACML) version 3.0 plus errata 01. Standard, OASIS, 2017.

[29] E. Yuan and J. Tong. Attributed Based Access Control (ABAC) for web services. In *Proceedings - 2005 IEEE International Conference on Web Services, ICWS 2005*, volume 2005, pages 561–569, 2005.

Article D

Chapter 10

Article D: Recipe Based Access Control in Modular Automation

Björn Leander, Aida Čaušević, Hans Hansson
MRTC Report, MDH-MRTC-333/2020-1-SE, Mälardalen Real-Time Research Centre, Mälardalen University, 2020

Abstract

In the emerging trend towards modular automation, a need for adaptive, strict access control between interacting components has been identified as a key challenge. In this article we discuss the need for such a functionality, and propose a workflow-driven method for automatic access control policies generation within a modular automation system.

The solution is based on recipes, formulated using Sequential Function Charts (SFC). The generated policies are expressed using Next Generation Access Control (NGAC), an Attribute Based Access Control (ABAC) standard developed by NIST. We provide (1) a definition of required policies for device-to-device interactions within a modular automation system, (2) an algorithm for automatic generation of access policies, (3) a formal proof of the correctness of this algorithm, and (4) an illustration of its use.

10.1 Introduction

Modular Automation (MA) [25] is an emerging technology within the process automation industry that promises to enable profitable operations, reduced time-to-market and shortened product life cycles [10]. Even though the technology is in its infancy, a number of pilot projects have been already carried out[1], along with a number of control system vendor implementations specifically targeting MA[2]. Within the chemical, pharmaceutical, and energy sectors there is an estimated 2030 market potential of approximately 12 billion euros for modular process automation equipment [26].

The technology suggested to be used in MA exhibits similar characteristics as solutions provided in the Industry 4.0 paradigm, namely interconnected service oriented devices, utilizing different connectivity capabilities, including wireless communication [20, 22]. The different entities within the systems are assumed to be highly heterogeneous and dynamic, and the architecture is expected to be modular, with different modules able to autonomously fulfill specific tasks, requiring only high level engineering to combine and re-combine modules to execute the complete production scheme. This allows a high level of customization and re-use of modules provided and possibly maintained by specialized vendors.

In these dynamic and flexible systems where communication paths are not pre-defined, and production schemes are ever-changing, it becomes difficult to detect malicious behaviour, at least between devices seen as legitimate. At the same time, the attack surface and complexity of the system is increasing, raising the risk of a legitimate device being compromised.

A compromised device, controlled by a malicious actor, may cause a significant economic damage for the factory owner, as well physical damage on e.g., humans, machinery or the environment. The impact may be direct, e.g., the opening of a valve may overfill a tank or turning on heating in an empty reactor may cause a fire. Impact could also be indirect, e.g., changing ratios of materials used to produce a medicine may render it harmful. The direct causes are usually mitigated by implementations of secondary safety measures, while indirect causes may be more difficult to detect and mitigate.

During the last years, there has been a steady trend of increasing amounts

[1] new.abb.com/life-sciences/references/modular-automation-solution-for-life-science-company-bayer-ag

[2] new.abb.com/news/detail/31671/plant-orchestration-and-pilot-application, www.festo.com/us/en/e/automation/industries/water-technology/modular-automation-id_4801/

of cyber-attacks on industrial control systems [19]. When analyzing who is attacking and why attacks occur against different targets, there is a number of standard categories [15, 8] used: Hobby hacker, Insider, Cyber-criminal, Hacktivist, Terrorist and Nation state. For attacks against industrial control systems, the two main categories with knowledge and capacity to perform targeted attacks are the Insider and the Nation State. However, any of the other categories can use an Insider to gain initial foothold, e.g., by social engineering, bribery or extortion. An Insider can hold deep knowledge of the system, credentials, as well as physical access to the system.

Applying strict and fine-grained access control according to the principle of *least-privilege* [17] is one of the major mechanisms able to protect against the threat from Insider attacks, by allowing access to operations or data only to privileged entities. It also increases the visibility of the malicious actor, as denied access control requests are typically monitored e.g., using a Security Information and Event Monitoring (SIEM) system [7]. However, using a strict access control at the lower layers in an automation system is quite uncommon. Historically, industrial automation systems have been built up using proprietary communication protocols, hard-wiring between controllers and IO, and the notion of an air-gapped network, i.e., no communication between the control network and the outside world. These assumptions on the technical solutions have meant that the pragmatic solution is to allow any legitimate device on the network to perform any action. With the advent of MA and Industry 4.0 none of these assumptions hold anymore, and therefore the practice of including a strict access control between devices in automation systems is of increasing importance.

Two of the main hurdles to introducing access control for machine-to-machine interactions in a MA system are the difficulty to express policy rules matching the dynamic behavior of the system, and the management effort required to uphold the policies in a timely and efficient manner. In relation to that, the following research questions are stated:

RQ1 How can access control policies be expressed to fulfill the principle of least-privilege for device to device interactions within a MA system?

RQ2 How can the effort related to access control policy management be minimized in a MA scenario?

In this paper we propose an approach providing answers to both these questions, by introducing a model-based method for generating access control policies from formalized recipe descriptions. We present a definition on re-

quired access control rules for recipe orchestration, and provide a formal proof showing that the algorithm produces rules in accordance with that definition. Moreover, we apply the algorithm on a simple example.

The remainder of this paper is structured as follows. Definitions are given in Section 10.2, including a formal definition of the requirements on privileges required during the recipe orchestration. In Section 10.3, an algorithm for access policy generation is described followed by an illustrative example in Section 10.4. Furthermore, we discuss the proposed solution and results in Section 10.5, compare it to relevant related work in Section 10.6, before making a few concluding remarks in Section 10.7.

10.2 Preliminaries

10.2.1 A Recipe definition using an SFC

The execution of a workflow in MA is described by a *recipe* with different processing steps, each containing a set of operations that one or more modules shall perform. A common format used to describe a recipe is through a Sequential Function Chart (SFC), which is currently used e.g., for batch processing in traditional process automation. Execution of a recipe is driven by a central unit, following the concept of orchestration of autonomous services [14]. SFC is a high-level Programmable Logic Controllers (PLC) language, defined within the IEC 61131 standard [6].

Let us consider an example of a simple MA setup described within the DIMA project [10]. In order to produce a specific product, different modules are combined and a process recipe is being formulated as follows:

1. A reactor is filled with three different materials in a specific ratio.
2. The reactor module mixes and heats the mixture, and maintains a fixed temperature for a specified amount of time.
3. The resulting mixture is distilled by a distilling module.
4. The distillate is further purified by a filtration module.
5. The product is packed into a container by a filling module.

The example can be formulated as a recipe using an SFC, as illustrated in Fig. 10.1. We assume that filtration and packaging can be executed in parallel, i.e., the packaging can start as soon as there is a sufficient amount of the final product available.

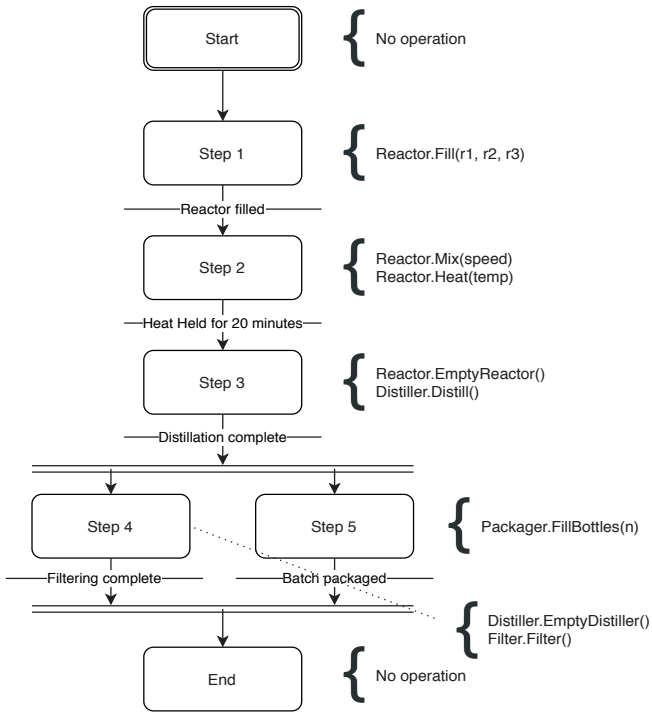


Figure 10.1: An example of a recipe expressed as an SFC

An SFC consists of steps and transitions. Each step in the recipe describes the operations relevant to perform in that step. Moreover, each step contains zero or more outward directed transitions (arcs) describing the conditions for continuing to the next step(s), i.e., a transition point to one or more subsequent steps. In the case of more than one step, the following steps are executed in parallel as soon as the condition annotated on the transition enabling that step is fulfilled. To join a parallel execution, two (or more) edges point to the same step. In such join-cases, conditions for all edges pointing at the same step must be fulfilled for it to be triggered. Moreover an SFC may contain loops (not included in Fig.10.1).

In general, operations described for a step contain code describing operations detailing the control logic of a step. The standard allows nested SFCs, so that a step can be described by another SFC. However, in MA recipe declaration, this description will most likely be vastly simplified, as the modules are expected to perform the low level control logic by themselves, based on high-level instructions executed by the orchestrator. For our description of an

SFC formulating a modular automation recipe, the important aspect is that one step contains zero or more module-related operations.

A recipe R is a pair (id, s_0), where id is a unique recipe identification, and s_0 is the initial step of an SFC $F = (S, s_0)$. $F.S$ is the set of all steps contained by the SFC F. A $step$ is defined as a triplet $step = (id, OP, T)$ where OP is a set of operations, T is set of transitions and id is a unique identifier for the step. Moreover, $op \in OP$ is described by a pair $op = (id, target)$, where id is a unique operation identification, and $target$ identifies the target module. A transition $t \in T$ is described by a pair $t = (c, steps)$ where c is a Boolean condition that must hold for the transition to be fired, and $steps$ is a set of one or more (parallel) steps to be activated by the transition.

For the approach of a policy generation algorithm presented in this article, the condition of a transition is not used. However, in future versions we envision using the conditions to more closely make the policy rules match the workflow of the recipe.

10.2.2 An NGAC graph definition

Access control is the practice of granting or denying a legitimate subject privileges to a requested resource [18]. An Access Control Model is a model for formally describing access control policies. Next Generation Access Control (NGAC) [5] is a NIST standardized access control model, based on the paradigm of Attribute Based Access Control (ABAC). In ABAC, attributes of the subject, resource and the environment are used to express the policies, as opposed to traditional models that are usually based mainly on the identity or role of the subject [24].

In the following, we provide a simplified description of NGAC, based on the work by Ferraiolo et al. [4], focusing to describe only those parts of the mechanism important for the purpose of this article. We exclude the details regarding prohibitions, while obligations will be briefly discussed later on in this article.

In NGAC, attribute assignments and privilege associations are described using a graph G. Subjects s, objects o, policy-classes pc and attributes a are modeled as vertices in the graph. Assignments of attributes to subjects, objects or policy-classes are modeled as directed edges ending at the assignment target. Assignments are also allowed between attributes, so that hierarchies of attributes can be formed. The assignment operation should be interpreted as containment, e.g, $o \rightarrow a$ means an object o is contained by an attribute

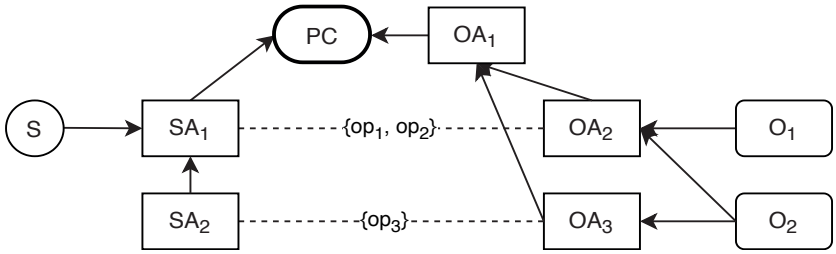


Figure 10.2: An example of an NGAC graph

a. Privileges to execute operations are modeled as associations between subject and object attributes (sa and oa, respectively) and described as a triplet: (sa, ops, oa) and visualized in the graph as an un-directional dashed line between the subject and object attributes, where ops is a set of operations.

Let us consider an example depicted in Fig. 10.2. It describes an NGAC-graph, where a subject S is assigned to attribute SA_1; objects O_1, O_2 are assigned to attribute OA_2. Between OA_2 and SA_1 there exists a privilege association for operations $\{op1, op2\}$. Furthermore, an object O_2 is assigned to attribute OA_3, and there exists a subject attribute SA_2 associated with OA_3 for operation $\{op_3\}$. Object attributes OA_2 and OA_3 are assigned to OA_1. OA_1 and SA_1 are both contained in policy class PC. Using the privilege association between SA_1 and OA_2 the operations op_1 and op_2 on objects O_1 and O_2 are granted to the subject S. However, an operation op_3 on O_2 is not granted, since no association can be made between the subject S and the object O_2 for the given operation. If, on the other hand, S would have been contained in attribute SA_2, then operations op_1, op_2, op_3 would have been allowed on O_2.

For operations on an NGAC graph, we will use a number of definitions from [5]. A short summary of these definitions and their meaning is provided in Table 10.1. In NGAC, the term ***user*** denotes the same entity as we denote ***subject***, therefore e.g., ua represents "user attribute", while we write "subject attribute" to maintain a consistent nomenclature within the article.

For operation op on object o executed by subject s, we say that (s, op, o) is a

Name	Description
PC	The set of all policy-classes in the graph.
ASSOCIATIONS	The set of all associations in the graph, one association being defined by a triple (sa, ops, oa).
ASSIGN$^+(x, y)$	There exists a series of assignments from x to y. Note that in the NIST standard [5], the notation used is $(x, y) \in$ ASSIGN$^+$.
CREATEOAINPC(oa, pc)	Create an object attribute with id oa and assign it to policy class pc. A call of this function implies that ASSIGN$^+(oa, pc)$ is fulfilled.
CREATEUAINPC(ua, pc)	Create an subject attribute with id ua and assign it to policy class pc. A call of this function implies that ASSIGN$^+(ua, pc)$ is fulfilled.
CREATEOAINOA(oa_1, oa_2)	Create an object attribute with id oa_1 and assign it to object attribute oa_2. A call of this function implies that ASSIGN$^+(oa_1, oa_2)$ is fulfilled.
CREATEUAINUA(ua_1, ua_2)	Create a subject attribute with id ua_1 and assign it to subject attribute ua_2. A call of this function implies that ASSIGN$^+(ua_1, ua_2)$ is fulfilled.
CREATEASSOC(sa, ops, oa)	Creates an association between subject attribute sa and object attribute oa with operations ops, i.e. ASSOCIATIONS $=$ ASSOCIATIONS $\cup \{(sa, ops, oa)\}$. We assume that consecutive calls using the same combination of sa and oa will update the set of operations for the association using a set-union function.

Table 10.1: NGAC-operations

privilege using the following definition:

$$\text{PRIVILEGE}(s, op, o) = \begin{cases} true \text{ if } \begin{cases} \forall pc \in \text{PC} : \text{ASSIGN}^+(o, pc) \wedge \\ (\exists (sa, ops, oa) \in \\ \text{ASSOCIATIONS} : op \in ops) \wedge \\ \text{ASSIGN}^+(s, sa) \ \wedge \ \text{ASSIGN}^+(o, oa) \ \wedge \\ \text{ASSIGN}^+(s, pc) \ \wedge \ \text{ASSIGN}^+(oa, pc)) \end{cases} \\ false \text{ otherwise} \end{cases} \tag{10.1}$$

Intuitively, this means that for for the privilege of s executing operation op on target object o to be granted for a policy-class pc containing o, there must exist an association between a subject attribute sa and an object attribute oa containing operation op, where s is assigned to attribute sa and o is assigned to attribute oa, and both s and oa are assigned to the policy class pc.

10.2.3 Definition of privileges required by a recipe orchestrator

Using the definitions introduced in Sections 10.2.1 and 10.2.2, we are able to define which access control privileges are required by a recipe orchestrator when a recipe formalized as an SFC is executed, following the principle of least privilege.

An orchestrator $subj$ is allowed to execute a step $step \in F.S$ for an SFC $F = (S, s_0)$ where the access control policies are described by an NGAC

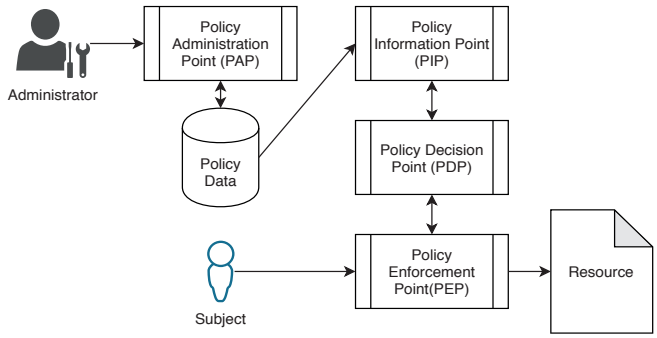


Figure 10.3: Access Control Architecture

graph by the following definition:

$$\textsc{Priv}_{\textsc{step}}(subj, step) = \begin{cases} true \text{ if } \forall op \in step.OP : \textsc{Privilege}(subj, op.id, op.target) \\ false \text{ otherwise} \end{cases}$$

Subsequently, for the orchestrator $subj$ to execute a recipe $R = (id, s_0)$, where SFC $F = (S, s_o)$, $\textsc{Priv}_{\textsc{step}}(subj, step)$ must be fulfilled for all steps in the SFC, i.e.,

$$\textsc{Priv}_{\textsc{recipe}}(subj, R) = \begin{cases} true \text{ if } \forall \ step \ \in \ F.S : \textsc{Priv}_{\textsc{step}}(subj, step) \\ false \text{ otherwise} \end{cases} \tag{10.2}$$

10.2.4 Access Control Architecture prerequisites

Policy enforcement is an important characteristic of an access control mechanism. Fig. 10.3 depicts a typical architecture that describes the entities involved in an access control enforcement architecture [4, 23, 11]. For the mechanism to work, there can be no other way for a subject to access a resource than through a Policy Enforcement Point (PEP). Therefore PEP must be kept close to the resource, typically running on the same device as the resource. After a privilege request is initiated, a PEP must ask a Policy Decision Point (PDP) for a decision defining whether the request shall be granted or not. To answer the request, a PDP must be able to query policy data through a Policy Information Point (PIP). Policy data is administered through the Policy Administration Point (PAP). The actual placement and implementation of these

policy interaction points will be of great importance, as it influences how well the access control mechanism functions and scales. In this article, we assume that an appropriate such architecture is in place. Another prerequisite for a secure access control is that identities of all involved entities can be trusted.

Secure authentication of entities can be achieved using a number of methods, including public key certificates. In this article we will assume that authenticity of identities are proven using some trusted mechanism.

10.3 Generating Access Control rules in NGAC using an SFC Recipe

As a recipe is activated and assigned to modules and an orchestrator, the access control policies prescribing which operations that a specific orchestrator is able to perform within the system shall also be updated. Similarly, deactivation of a recipe shall remove privileges exclusively granted through that recipe.

In this article we are focusing on the interactions between orchestrators and modules in an MA system. To define these interactions, we propose to use the recipe as a basis to formalize and automate access control rule generation. Based on these rules, it is possible to grant only those privileges prescribed by the processing needs. We propose to use the recipe as a model, further used to derive detailed policy rules expressed as ABAC policies according to the NGAC specification.

In this section we introduce an algorithm to enable automatic generation of access control policies. The algorithm takes a formalized SFC model as input and as result updates an NGAC graph with access privilege information.

For each step in an SFC, zero or more operations are allowed to be executed by the orchestrator on the target modules. Therefore it is logical to, in terms of NGAC attributes, think of the SFC steps as subject attributes. Based on them, privileges will be granted to the orchestrator by associations to a respective target module specific attribute.

Algorithm 1 PolicyGeneration(R, pc)

```
1: function GenerateStepPolicies(step, R_id)
2:     if step.OP ≠ {} then
3:         mod := mod_id(R_id)
4:         orch := orch_id(R_id)
5:         CreateUAinUA(orch, step.id)
6:         for all op ∈ step.OP do
7:             targ := target_id(op.target, R_id)
8:             CreateOAinOA(targ, mod)
9:             CreateAssoc(step.id, op.id, targ)
10:        end for
11:    end if
12: end function
13:
14: function VisitStep(step, R_id)
15:    if ¬visited(step) then
16:        visit(step)
17:        GenerateStepPolicies(step, R_id)
18:        for all t ∈ step.T do
19:            for all sub_step ∈ t.steps do
20:                VisitStep(sub_step, R_id)
21:            end for
22:        end for
23:    end if
24: end function
25:
26: begin algorithm
27:    orch := orch_id(R.id)
28:    mod := mod_id(R.id)
29:    CreateOAinPC(mod, pc)
30:    CreateUAinPC(orch, pc)
31:    VisitStep(R.s_0, R.id)
32: end algorithm
```

In the algorithm, we use the functions described for an NGAC graph in Table 10.1, together with the following functions:

- mod_id(R_{id}) - returns a unique attribute id for all modules being orchestrated by the recipe, based on the recipe id.
- orch_id(R_{id}) - returns a unique attribute id for the orchestrator of the recipe based on the recipe id.
- target_id$(target, R_{id})$ - returns a unique attribute id for a specific

module, based on the recipe id and the target id as used in the recipe.

Algorithm 1, PolicyGeneration, is called using a recipe R, and a policy class pc as arguments. The policy class pc must be predefined, and could e.g., be used to keep together all the policies related to control of modules. Unique attributes for module mod and orchestrator $orch$ are generated, based on the recipe identification $R.id$. The attribute mod will be common for all modules related to the recipe $R.id$, and the attribute $orch$ will be used for the orchestrator of the recipe $R.id$. As can be seen, $\text{Assign}^+(mod, pc)$ and $\text{Assign}^+(orch, pc)$ are the major result of the algorithm. Finally, function VisitStep is called using the initial step of the recipe, $R.s_0$, as input.

In VisitStep, the function visited(S) and method visit(S) are used to be able to determine if policies are already generated for the specific step. If the step has not been previously visited, function GenerateStepPolicies is called. For all the transitions $t \in step.T$, all $sub\_step \in t.steps$ are used as arguments for calls to VisitStep, to ensure policy generation for steps following a transition from the input parameter step.

In GenerateStepPolicies, if there are any operations related to the step, then (1) a subject attribute representing the step in the SFC is created based on the unique identification of the step, (2) $orch$ is assigned to it, i.e., $orch \rightarrow step.id$, (3) for all operations $(target, op)$ in the step, an attribute $targ$ is created for the target module unique within the recipe, such that $targ \rightarrow mod$, and (4) an association is created between attributes $step.id$ and $targ$ such that $\exists(step.id, ops, targ) \in \text{Associations} : op \in ops$.

10.3.1 A proof of algorithm correctness

In the following we provide a proof that by induction shows that the algorithm will create access control policies fulfilling the relationship as defined in Section 10.2.3, i.e., that $\text{PRIV}_{\text{RECIPE}}(subj, R)$ is fulfilled. The proof is divided into three lemmas and a proof of the main theorem based on the lemmas.

In the proof we rely on the transitive property of the Assign^+ relation, i.e.,:

$$\text{Assign}^+(a, b) \wedge \text{Assign}^+(b, c) \implies \text{Assign}^+(a, c) \tag{10.3}$$

Theorem 1. *Algorithm 1 will create policies fulfilling definition $\text{Priv}_{\text{RECIPE}}(subj, r)$ for a recipe $R = (id, s_0)$, an orchestrator $subj$, and a set of target modules T_m, using an NGAC graph containing the policy class pc, under the following assumptions:*

$$\textsc{Assign}^{+}(subj, \textsc{orch\_id}(R.id)) \tag{10.4}$$

$$\forall t \in T_m : \textsc{Assign}^{+}(t, \textsc{target\_id}(t, R.id)) \tag{10.5}$$

$$\exists! pc \in \text{PC} : \forall t \in T_m : \textsc{Assign}^{+}(t, pc) \tag{10.6}$$

Intuitively, the theorem states that Algorithm 1 provides access control policies fulfilling the principle of least privilege with regards to recipe orchestration.

The first assumption described (Eq. 10.4) states that the orchestrator $subj$ will need to be assigned to attribute $\textsc{orch\_id}(R.id)$. The second assumption (Eq. 10.5) states that the modules being used in recipe orchestration will have to be assigned to a unique attribute for the combination of the recipe and target id. Both of these assumptions should be fulfilled during recipe activation, as part of the operation engineering phase. Therefore these assumptions are necessary and valid.

The third assumption (Eq.10.6) states that there is exactly one policy class for privileges related to the target modules being orchestrated. As the purpose of a policy-class is to organize and distinguish between distinct types of policies [5], it is reasonable to make this assumption. This is indicated by the first part of the privilege definition (Eq. 10.1): $\forall pc \in \text{PC} : \text{ASSIGN}^{+}(o, pc)$. It follows that for a multi-policy scenario where one object is contained by more than one policy-class, for any privilege to be granted in relation to that object, associations must be present in all of the containing policy-classes.

Lemma 1 (Policy generation for a single SFC step). *Function GenerateStepPolicies (Alg. 1, Ln. 1) generates access control policies fulfilling* $\textsc{Priv}_{\textsc{step}}(subj, step)$ *for any step in an SFC used as parameter, given that:*

$$\textsc{Assign}^{+}(subj, pc) \tag{10.7}$$

$$\textsc{Assign}^{+}(\textsc{mod\_id}(R_{id}), pc) \tag{10.8}$$

Proof. By induction on $op \in step.OP$.
Base case: For a SFC step $step$ with $step.OP = \{\}$, a call to GenerateStepPolicies will fulfill $\textsc{Priv}_{\textsc{step}}(subj, step)$.

In this case the proof is trivial. No policy elements will be created. Hence, there are no operation in $step.OP$, and $\textsc{Priv}_{\textsc{step}}(subj, step)$ is vacuously true.

Induction hypothesis: Assume that for a step $step$, GENERATESTEPPOLICIES will grant privileges fulfilling $\text{PRIV}_{\text{STEP}}(subj, step)$.
Induction: Let $step'$ contain operations $step'.OP = step.OP \cup \{(op', t')\}$.

As $step'.OP \neq \{\}$, attribute assignments and associations will be provided according to the following:

1. From Alg. 1, Ln. 5

$$\begin{aligned}\text{CREATEUAINUA}(orch, step'.id) \implies \\ \text{ASSIGN}^{+}(orch, step'.id) \\ \implies \text{ASSIGN}^{+}(subj, step'.id)\end{aligned} \quad (10.9)$$

since $\text{ASSIGN}^{+}(subj, \text{ORCH\_ID}(R.id))$ is in our initial assumption (Eq. 10.4), and $orch \equiv \text{ORCH\_ID}(R.id)$.

2. For the additional operation (op', t') (Alg. 1, Ln. 8):

$$\begin{aligned}\text{CREATEOAINOA}(targ, mod) \implies \\ \text{ASSIGN}^{+}(targ, mod) \implies \text{ASSIGN}^{+}(targ, pc)\end{aligned} \quad (10.10)$$

due to $\text{ASSIGN}^{+}(\text{MOD\_ID}(R_{id}), pc)$ in the assumptions of this lemma, and $mod \equiv \text{MOD\_ID}(R_{id})$. Furthermore, $targ \equiv \text{TARGET\_ID}(t', R_{id})$ according to the initial assumptions (Eq. 10.5), and therefore $\text{ASSIGN}^{+}(t', targ)$ is fulfilled.

3. From Alg. 1, Ln. 9:

$$\begin{aligned}\text{CREATEASSOC}(step'.id, op', targ) \equiv \\ \exists(step'.id, ops, targ) \\ \in \text{ASSOCIATIONS} : op' \in ops\end{aligned} \quad (10.11)$$

It then follows by stated assumptions (Eq. 10.6, 10.7):

$$\begin{aligned}\exists(step'.id, ops, targ) \in \text{ASSOCIATIONS} : op' \in ops \\ \wedge \text{ASSIGN}^{+}(subj, step'.id) \wedge \text{ASSIGN}^{+}(t', targ) \\ \wedge \text{ASSIGN}^{+}(subj, pc) \wedge \text{ASSIGN}^{+}(targ, pc)\end{aligned} \quad (10.12)$$

As we assume that there exists exactly one pc for operations related to target modules (Eq. 10.6), $\text{PRIVILEGE}(subj, op', t')$ is true according to Eq. 10.1. Since $\text{PRIV}_{\text{STEP}}(subj, step)$ is satisfied according to the inductive assumption and $step'.OP = step.OP \cup \{(op', t')\}$, we have shown that also $\text{PRIV}_{\text{STEP}}(subj, step')$ is true.

Base case + induction shows that for any SFC step $step$ where GENERATESTEP-POLICIES$(step, ...)$ is called, PRIV$_{\text{STEP}}(subj, step)$ will be fulfilled, under given assumptions. ■

Lemma 2 (Policy generation for Visited steps). *A step p visited by procedure VISITSTEP$(p, ...)$, will imply that PRIV$_{STEP}(subj, p)$ is fulfilled.*

Proof. VISIT(p) will set the VISITED(p). Furthermore, from Alg. 1, Ln. 17

$$\text{GENERATESTEPPOLICIES}(p, R_{id}) \implies \text{PRIV}_{\text{STEP}}(subj, p) \tag{10.13}$$

according to Lemma 1. ■

Lemma 3 (Policy generation for an SFC). *For a recipe $R = (R_{id}, s_0)$ where SFC $F = (S, s_0)$, a call to function VISITSTEP$(s_0, ...)$ will generate policies such that PRIV$_{RECIPE}(subj, R)$ is fulfilled.*

Proof. By induction on $step \in F.S$
Base case: For a recipe $R = (R_{id}, s_0)$ where SFC $F = (S, s_0)$ with $F.S = \{s_0\}$, a call to VISITSTEP$(s_0, ...)$ will generate policies such that PRIV$_{\text{RECIPE}}(subj, R)$ is fulfilled. By Lemma 2 it follows that PRIV$_{\text{STEP}}(subj, s_0)$ is fulfilled. Given that $F.S = \{s_0\}$ Eq. 10.2 is also fulfilled, which proves the base case.
Induction hypothesis: Assume that for a Recipe $R = (R_{id}, s_0)$ where SFC $F = (S, s_0)$ and contains step $step_i \in S$, procedure VISITSTEP using s_0 as parameter will produce policies fulfilling the definition in Eq. 10.2.
Induction: Let recipe $R' = (R_{id}, s_0)$ where SFC F' being F extended with one additional step $step_{i+1} \neq s_0$ such that $F'.S = F.S \cup \{step_{i+1}\} \wedge \exists trans \in step_i.T : step_{i+1} \in trans.steps$.

For $step_i$, VISITSTEP(sub_step, G, R_{id}) implies that VISITSTEP will be called for $step_{i+1}$, since $\exists\ trans \in step_i.T : step_{i+1} \in trans.steps$. According to Lemma 2, this implies that PRIV$_{\text{STEP}}(subj, step_{i+1})$ is true. For F, we have that $\forall step \in F.S$: PRIV$_{\text{STEP}}(subj, step)$. As $F'.S = F.S \cup \{step_{i+1}\}$, the following holds:

$$\text{PRIV}_{\text{STEP}}(subj, step_{i+1}) \wedge \forall step \in F.S : \text{PRIV}_{\text{STEP}}(subj, step) \implies \forall step \in F'.S : \text{PRIV}_{\text{STEP}}(subj, step) \tag{10.14}$$

which is according to $Eq.\ 10.2$ is equivalent to PRIV$_{\text{RECIPE}}(subj, R')$.

Base case + induction proves that for any recipe $R = (R_{id}, s_0)$ where SFC $F = (S, s_0)$, a call to function VISITSTEP will fulfill the definition in Eq. 10.2, if the assumptions in Lemma 1 holds. ■

Recalling Theorem 1, we will now show that for any recipe $R = (R_{id}, s_0)$, Algorithm 1 will generate policy elements fulfilling $\text{PRIV}_{\text{RECIPE}}(subj, R)$

Proof of Theorem 1.

$$\text{CREATEOAINPC}(mod, pc) \implies \text{ASSIGN}^{+}(\text{MOD\_ID}(R_{id}), pc) \tag{10.15}$$

and, from the initial assumption (Eq.10.4),

$$\begin{aligned}\text{CREATEUAINPC}(orch, pc) \implies \text{ASSIGN}^{+}(orch, pc),\\ \text{ASSIGN}^{+}(subj, \text{ORCH\_ID}(R.id)) \wedge \text{ASSIGN}^{+}(orch, pc)\\ \implies \text{ASSIGN}^{+}(subj, pc)\end{aligned} \tag{10.16}$$

Thereby, both stated assumptions in Lemma 1 are fulfilled. Furthermore, $\text{VISITSTEP}(R.s_0, R.id)$. is called. Together, this imply that Eq. 10.2 is fulfilled according to Lemma 3.

Consequently, we have proved that the initial theorem is correct. The proposed algorithm will generate access control policies fulfilling the definition in Eq. 10.2, required for an orchestrator $subj$ to execute a recipe $R = (R_{id}, s_0)$, i.e., $\text{POLICYGENERATION}(R, pc) \implies \text{PRIV}_{\text{RECIPE}}(subj, R)$. ■

10.4 Proposed algorithm exemplified

Let us consider using the proposed algorithm on the example of the recipe described by the SFC in Fig. 10.1. For readability reasons, in this example we use a string representations for attribute and entity IDs. In reality these will most likely be numeric IDs. As an input to the algorithm we use the SFC and a policy class, which is assumed to already be existing in the NGAC-graph. We annotate them as "Module Control Policies" as ID for the policy class.

In the main part of the algorithm, firstly two unique attributes will be generated, one for the orchestrator ("Recipe ID Orchestrator") and one for the modules ("Recipe ID Module"), see lines 26-30 in ALGORITHM 1. After that the function VISITSTEP is called using the step Start of the recipe as input, along with the id of the recipe.

In function VisitStep the step Start is marked as visited (lines 15-16 in Algorithm 1) and then the GenerateStepPolicies function is called, using the step as input parameter. As the starting step contains no operations, nothing happens in this first call to GenerateStepPolicies (condition on line 2 in Algorithm 1 is false). On line 20 "Step 1" is used as an input to a recursive call to VisitStep, which leads to a call to GenerateStepPolicies using "Step 1" as an input.

Since there is an operation related to Step 1, a subject attribute related to the step is created ("Recipe ID Step 1") on line 5 such that "Recipe ID Orchestrator"→"Recipe ID Step 1", followed by lines 7-8 where an object attribute is created for the module (in this case "Recipe ID Reactor"), such that "Recipe ID Reactor"→"Recipe ID Module". On line 9, an association between the attribute "Recipe ID Step 1" and "Recipe ID Reactor" is created, containing the operation "Fill". In this way all the steps of the SFC are iterated, thus creating the NGAC sub-graph related to this specific recipe.

The illustration in Fig. 10.4 depicts the NGAC sub-graph related to this policy, after recipe activation. The gray area in the graph represents the results of executing our algorithm. The module and orchestrator assignments to the respective attributes are part of a recipe activation, and the policy class is assumed to be existing prior to the execution of the algorithm. We omit the details regarding the assignment of modules and orchestrator to the respective attribute in the proposed algorithm. The physicals modules that are a part of the manufacturing scheme must be selected by the operational engineer upon recipe activation. We assume that there will be a simple way to match the physical module ID with the representative ID used in the recipe.

As can be seen, each of the steps from the original SFC are represented by subject attributes in the NGAC-graph, and all the individual modules utilized in the SFC are given as object attributes. The privileges are described as associations between the step-, and module-attributes, e.g., for Step 3, there is one association to the Reactor-attribute, granting an operation "EmptyReactor", and one association to the Distiller-attribute, granting an operation "Distill".

10.5 Discussion

The suggested approach of restricting access control policies based on the recipe description would effectively prevent any entity to perform actions on modules outside of active recipes, mitigating the effects of a compromised or faulty device with regards to execution of operations. Depending on the

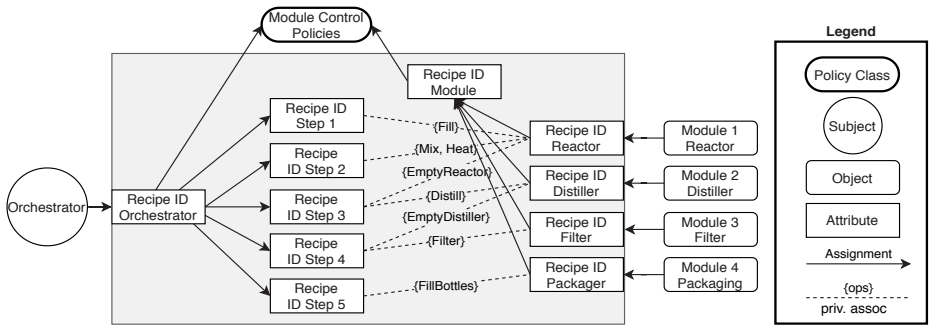


Figure 10.4: Example of NGAC policy and attribute setup for the recipe described in Fig. 10.1.

implementation of the authorization enforcement layer, it could also improve the resilience against a Denial of Service (DoS) attack against a module or the orchestrator, as processing of unauthorized requests could be minimized, i.e., processing of malicious requests can be skipped if it can easily be determined that they are unauthorized. Furthermore, failed authorization is usually captured by audit logging and can be visualized in a SIEM system, increasing the visibility of the attacker. The approach would also provide an access control model supporting the concept of module reuse, which is one of the main objectives of MA. By automatically generating the access control policies from already existing engineering data, the management effort related to sustaining the rules in accordance with the least privilege principle is minimized.

A difference between NGAC and other ABAC implementations, e.g., eXtensible Access Control Markup Language (XACML) [23], lies in the fact that an attribute does not represent a named property that can hold different typed values. This results in creation of several "synthetic" attributes, i.e., attributes that are not naturally associated with a subject or object. For example, as a result of the policy generation algorithm, the need for unique attributes for each combination of recipe and module yields a large number of attributes that can only be used in the context of execution of a specific recipe. A more natural concept would be to have a general attribute for all modules representing the recipe that the module currently is assigned to. An evaluation of the execution cost for such a growing number of attributes in NGAC should be performed, if considering using this approach in a scaled up scenario with real time requirements.

There are aspects of SFC recipes that cannot be captured by the suggested

method for policy generation, e.g., related to the difficulty to express transitions between steps. Another aspect is the actual logic within one step of the SFC. There may be IF-conditions or loops that surrounds the module operations with additional logic, something not captured by the access control logic. A third aspect are parameters used for operations. A parameter set using a malicious value in an otherwise valid function call could have a harmful effect on the system.

Our suggested approach uses only positive grant policies, since they provide a natural way of describing the execution of an SFC. We do, however, not claim that this is always the best way of describing all kinds of recipe orchestration policies. There could be scenarios where combinations of grants and prohibitions provide a better solution, e.g., if a specific system state should prohibit an execution. For scenarios where policy evaluation leads to conflicting results, NGAC will always use the most restrictive outcome.

Despite these shortcomings, we see our approach as a potential mitigation against compromised devices in automation systems. Fine-grained access controls between devices is a useful additional layer of security which is not present at the moment, neither in traditional systems for process automation, nor in the current frameworks describing MA architectures.

10.5.1 On recipe activation

As mentioned, the module and orchestrator attribute assignments are omitted from the algorithm. In the following we provide some rationale behind that decision. First, as there may be considerable delay between the moment of completed integration engineering, in which recipe formulation is one part, and the start of production, there is a need for the graph to be created without granting any privileges. If an orchestrator attribute is assigned already at this point, the principle of least privilege would not be followed. Second, the generated graph can be used as a template. When e.g., increasing the production by adding additional production lines, there will be no need to generate new access control policies, instead the policies generated for the recipe can be reused as a template. Third, this approach allows for integration engineering using MTPs without the presence of the physical modules in the system, i.e., recipe formulation could be completed before a module procurement.

An alternative approach would be to not generate any part of the access control graph at all until the recipe is activated. This approach could be beneficial as the matching to modules and subject could be done with a minimal extension of the algorithm, and the total NGAC access control graphs would not be

burdened with "unused" parts related to recipes not actively in use. However, there might be potential issues related to who has the privileges to perform the administrative operations on access control policies. An attribute assignment and provisioning is usually done locally, while policy administration is done centrally, in the same way as recipe activation and supervision is done by an operation engineer, while recipe formulation is done by an integration engineer.

10.5.2 On recipe de-activation / decommissioning

When a recipe should no longer be used in the MA system, the question arises about the best way to dispose it, such that the privileges granted under the recipe are no longer active, but the actions performed during the recipe lifetime still can be explainable on review. One approach could be to remove all the attributes and associations related to the recipe from the NGAC graph. Another approach would be to keep the entire generated part of the graph, and only remove the attribute associations from the orchestrator and modules. The best choice depends on the expected life-cycle of a recipe.

10.5.3 On temporal policies and obligations

In the presented NGAC policy generation algorithm, the task transitions as described by the SFC are not at all considered. This is a violation against the principle of least privilege, since the orchestrator will be allowed to perform any of the operations prescribed by the SFC at any point in time, regardless of the current working step in the recipe. The use of *obligations* may be a way around this shortcoming.

Obligations in NGAC are described by a tuple (ep, r), informally expressed as "**when** ep **do** r", where ep is an event pattern and r is a response, containing one or more administrative operations. One example of an obligation in this context could be:
when *Orchestrator succesfully has performed Fill on Reactor* **do** *remove assignment of Orchestrator to Step 1, assign Orchestrator to Step 2.*
However, the obligations in NGAC are limited to describing policy-related events, i.e., there is no way of telling that the Orchestrator actually performed the Fill-operation on Reactor, only that the Orchestrator requested and has been granted (or denied) the right to perform the operation. Therefore, based on the current knowledge, the workflow characteristics of modular automation cannot be modeled using obligations in NGAC.

An alternative way of driving the workflow model would be to have an external

entity assigning and de-assigning the recipe step attributes to the orchestrator following the SFC state model. Such a scheme would fulfill the least-privilege principle, but would defeat the purpose of having an orchestrator being responsible for driving the state model for the recipe. This kind of solution is however common in e.g., safety controllers, where a secondary safety module receive the same input as the primary controller, performs the same logic and compares the resulting output, forcing the controller to a safe state on deviating results.

10.6 Related Work

Workflows as a basis for access control is discussed in several publications related to business process modeling and Process-Aware Information Systems (PAIS). A review of security related to PAIS is provided by Leitner et al. [12]. Knorr [9] discusses the use of workflows modeled as Petri-nets in an access control enforcement engine. Domingos et al. [3] suggest an access control model for adaptive workflows, based on RBAC. These works relate to our approach in the use of formalized workflow models as a basis for authorization, while the difference lies in the application domain, where the PAIS typically is implemented as a part of a business process system, e.g., for document handling or similar, whereas our approach aims at industrial control systems.

Task-based authorization control (TBAC), by Thomas et al. [21], is Access Control model aiming at achieving similar objectives as the approach presented in this paper, i.e., limit access control to a just-in-time and need-to-do basis, following task descriptions. Also in this field, the target applications are, e.g., for transaction management- and information management-systems. Furthermore, this field of research has not materialized in any generally accepted standards, and there are no well established reference implementations available.

Ruland et al. [16] describe an access control system for smart energy grids and similar IACS. The system works in two stages, the first one is based on a limited set of policies expressed in XACML, the second stage uses knowledge about behavior of the system to prevent actions outside defined boundaries, to maintain safety properties of the system. This approach is similar to the one we suggest, as the expected behavior of the system is used as basis to formulate the secondary stage policies. However, the supported use cases are rather static, and there is no effort toward automation of policy formulation. Nevertheless, the idea of separating the privilege inference in several stages could be interesting, especially for real-time sensitive applications.

In the field of Model-Driven Security (MDS), originating from Model Driven Architecture, there is a body of research related to the design of secure systems, with regards to modeling, analysis as well as model transformation. Basin et al. [2], summarizes a lot of that work. The focus of MDS is mainly on the design phase for including security specific models when realizing a system architecture, by e.g., defining modeling languages for access control rules [13]. Most of MDS research is, with regards to access control, focused on the RBAC-model, there are however some examples utilizing attribute based access control; including Alam et al. [1] that describe a MDS approach for SOA, with XACML as policy expression language, and Lang et al. [11] that present a proximity-based access control model originating from the ABAC model, where the low-level policies are generated based on high-level policies described in natural language. An important argument from [11] is that: for ABAC in general, MDS is a requirement, as the low-level policy descriptions are so complex they cannot be managed without some amount of automation. To the best of our knowledge, there are no examples of MDS applied to policy automation in systems having properties similar to the ones of MA. In particular, we are not aware of any work covering a system where the policies needs to change dynamically, as required by the orchestration of modules in a MA system.

10.7 Conclusions

In this work, a method for automated access policy generation in the context of MA is presented. The policies are generated using recipes expressed in SFCs, which is an industry standard for PLC programming in the 1131 family. The resulting policies are described in the format of an NGAC sub-graph. With this work we have shown that efficient policy generation is possible in an MA system without any additional work being performed by engineering personnel. Using this algorithm in an industrial system would increase the system overall security by decreasing the maneuverability and increasing the visibility of a compromised device.

Recalling the initially stated research questions: RQ1 is related to how to express policies. As an answer we have provided a definition applicable to policies expressed using the NGAC model. RQ2 relates to minimizing the management effort related to access policy formulation in an MA system. The presented algorithm is one answer to that, describing how to use an available workflow model to automate the policy generation without the need for additional engineering efforts.

As future work we envision creation of an experimental setup allowing simulation of an MA system, including both integration and operational engineering, which will contain a full access control enforcement architecture using NGAC as policy engine. This would be one way to further confirm the results in this article, with regards to scalability. We also plan to further investigate mechanisms to support more fine-grained workflow-related characteristics of MA.

Using XACML instead of NGAC to express policy rules is another natural continuation of this work, trying to evaluate if policy generation is feasible in that framework. As XACML allows for valued and typed attributes, the policy generation may in that context not need the same amount of synthetic attributes.

Moreover, automated access-policy generation is also of interest in wider domains than MA, e.g., in smart manufacturing and other dynamic and flexible systems requiring fine grained access control policies. Extending our results into these domains are possible directions for further research.

Acknowledgements

This work is supported by the industrial postgraduate school Automation Region Research Academy (ARRAY, funded by the Knowledge Foundation) and ABB. The authors would like to acknowledge Tomas Lindström for his valuable feedback.

Bibliography

[1] M. Alam, R. Breu, and M. Hafner. Model-driven security engineering for trust management in SECTET. *Journal of Software*, 2(1):47–59, 2007.

[2] D. Basin, M. Clavel, and M. Egea. A decade of model-driven security. In *Proceedings of the 16th ACM Symposium on Access Control Models and Technologies*, SACMAT '11, page 1–10, New York, NY, USA, 2011. Association for Computing Machinery.

[3] D. Domingos, A. Rito-Silva, and P. Veiga. Authorization and access control in adaptive workflows. In *European Symposium on Research in Computer Security*, pages 23–38. Springer, 2003.

[4] D. Ferraiolo, R. Chandramouli, R. Kuhn, and V. Hu. Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC). pages 13–24, 2016.

[5] D. Ferraiolo, S. Gavrila, and W. Janse. Policy Machine: Features, Architecture and Specification. White paper, NIST, October 2015.

[6] IEC 61131-3:2013 Programmable Controllers - Part 3: Programming Languages. Standard, IEC, 2013.

[7] IEC 62443 security for industrial automation and control systems. Standard, Internation Electrotechnical Commission, Geneva, CH, 2009-2018.

[8] E. D. Knapp and J. T. Langill. *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems.* Syngress, 2014.

[9] K. Knorr. Dynamic access control through petri net workflows. In *Proceedings 16th Annual Computer Security Applications Conference (ACSAC'00)*, pages 159–167. IEEE, 2000.

[10] J. Ladiges, A. Fay, T. Holm, U. Hempen, L. Urbas, M. Obst, and T. Albers. Integration of modular process units into process control systems. *IEEE Transactions on Industry Applications*, 54(2):1870–1880, March 2018.

[11] U. Lang and R. Schreiner. Proximity-based access control (PBAC) using model-driven security. In H. Reimer, N. Pohlmann, and W. Schneider, editors, *ISSE 2015*, pages 157–170, Wiesbaden, 2015. Springer Fachmedien Wiesbaden.

[12] M. Leitner and S. Rinderle-Ma. A systematic review on security in process-aware information systems – constitution, challenges, and future directions. *Information and Software Technology*, 56(3):273 – 293, 2014.

[13] T. Lodderstedt, D. Basin, and J. Doser. SecureUML: A UML-based modeling Language for model-driven security. In *International conference on model engineering, concepts and tools*, 2002.

[14] C. Peltz. Web services orchestration and choreography. *Computer*, 36(10):46–52, Oct 2003.

[15] M. Rocchetto and N. O. Tippenhauer. On attacker models and profiles for cyber-physical systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9879 LNCS:427–449, 2016.

[16] C. Ruland and J. Sassmannshausen. Access Control in Safety Critical Environments. In *Proceedings - 12th International Conference on Reliability, Maintainability, and Safety, ICRMS 2018*, pages 223–229. IEEE, 2018.

[17] J. Saltzer and M. Schroeder. The Protection of Information in Computer Systems. In *proceedings of the IEEE*, volume 63, pages 1278–1308, September 1975.

[18] R. S. Sandhu and P. Samarati. Access control: Principles and Practice. *IEEE Communications Magazine*, 32(September):40–48, 1994.

[19] J. Slowik. Evolution of ICS Attacks and the Prospects for Future Disruptive Events. Technical report, 2017.

[20] K.-d. Thoben, S. Wiesner, and T. Wuest. "Industrie 4.0" and Smart Manufacturing – A Review of Research Issues and Application Examples. *International Journal of Automation Technology*, (January), 2017.

[21] R. K. Thomas and R. S. Sandhu. Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented authorization management. In *Database Security XI*, pages 166–181. Springer, 1998.

[22] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. V. Vasilakos. Software-defined industrial internet of things in the context of industry 4.0. *IEEE Sensors Journal*, 16(20):7373–7380, Oct 2016.

[23] eXtensible Access Control Markup Language (XACML) version 3.0 plus errata 01. Standard, OASIS, 2017.

[24] E. Yuan and J. Tong. Attributed Based Access Control (ABAC) for web services. In *Proceedings - 2005 IEEE International Conference on Web Services, ICWS 2005*, volume 2005, pages 561–569, 2005.

[25] ZVEI—German Electrical and Electronic Manufacturers' Association. Module-based production in the process industry—effects on automation in the "industrie 4.0" environment. White Paper, Frankfurt, 2015.

[26] ZVEI—German Electrical and Electronic Manufacturers' Association. Process INDUSTRIE 4.0: the age of modular production. White Paper, Frankfurt, 2019.

MÄLARDALEN UNIVERSITY
SWEDEN

Address: P.O. Box 883, SE-721 23 Västerås. Sweden
Address: P.O. Box 325, SE-631 05 Eskilstuna. Sweden
E-mail: info@mdh.se **Web:** www.mdh.se

ISBN 978-91-7485-478-7
ISSN 1651-9256