Automated Model-based Attack Tree Analysis using HiP-HOPS

Declan Whiting, Ioannis Sokoros, Yiannis Papadopoulos, Gilbert Regan, and Eoin O'Carroll

University of Hull, Faculty of Computer Science and Engineering,
Cottingham Road, Hull HU6 7RX, United Kingdom
{D.Whiting-2018,I.Sorokos,Y.I.Papadopoulos}@hull.ac.uk,
Dundalk Institute of Technology,
Dublin Road, A91 K584 Dundalk, Ireland,
gilbert.regan@dkit.ie,
Portable Medical Technology,
41/42 High Street, V93 T8K7 Kilarney, Ireland,
eoin@portablemedicaltechnology.com

Abstract. As Cyber-Physical Systems (CPS) grow increasingly complex and interact with external CPS, system security remains a non-trivial challenge that continues to scale accordingly, with potentially devastating consequences if left unchecked. While there is a significant body of work on system security found in industry practice, manual diagnosis of security vulnerabilities is still widely applied. Such approaches are typically resource-intensive, scale poorly and introduce additional risk due to human error. In this paper, a model-based approach for Security Attack Tree analysis using the HiP-HOPS dependability analysis tool is presented. The approach is demonstrated within the context of a simple web-based medical application to automatically generate attack trees, encapsulated as Digital Dependability Identities (DDIs), for offline security analysis. The paper goes on to present how the produced DDIs can be used to approach security maintenance, identifying security capabilities and controls to counter diagnosed vulnerabilities.

Keywords: attack trees, digital dependability identities, HiP-HOPS

1 Introduction

Cyber-Physical Systems (CPS) enhance traditional physical engineering systems with computational, often networked, control. CPS applications of particular importance are those found in domains of critical societal impact such as healthcare, transportation, energy, manufacturing and infrastructure control. Such applications offer considerable benefits in terms of enabling new capabilities, such as distributed control in traditionally centralized systems, e.g. power grids. Another potential benefit is improved efficiency, as semi-automatic, automatic and autonomous control can reduce human input and error and identify resource-optimal system behavior. In this respect, examples include autonomous control

of vehicles and smart structures. The European Commission's Smart CPS programme, part of the Horizon 2020, is indicative of their importance[1]. A more in-depth discussion of CPS considerations, requirements and potential solutions can be found in [14], [19].

In the aforementioned domains, safety is a key concern, as the implication of CPS failures could be catastrophic to the well-being of affected societies and the environment. As CPS combine both physical and digital aspects, they inherit the traditional concerns of reliability of their physical components impacting safety due to mechanical and/or development failure. However, with the introduction of digital control and network communication, CPS operation is also subject to security risks. Such risks are not necessarily in themselves novel, as they originate from digital technologies and infrastructure subject to extensive use and research. However, the complexity and novel internal and external interactions of CPS, coupled with the typical safety concerns mentioned previously, aggravates the impact of potential security attacks and necessitates rigorous treatment to mitigate the associated risks [6], [18].

Security concerns are addressed in highly variable methods in practice, depending on the application domain. Methods of systematic analysis, validation and verification can be employed to produce guarantees of system robustness against security attacks [7],[4],[20].

Tackling nominal system development alongside safety, reliability, security and, more generally, dependability concerns requires alignment of requirements elicitation and allocation, design and implementation activities with dependability assessment and assurance activities. When the above activities are not properly synchronized and dependent information is shared inaccurately or with delay, the associated discrepancy can cause further modification of the developed system later in the development life cycle, introducing much higher costs or even failure to appropriately identify and address critical system risks. Model-based dependability analysis is a paradigm that evolved from model-based design, centralizing both nominal and dependability-related development activities around a common, shared system model. The common model enables efficient and frequent synchronization across both tracks of development. As the models involved are also typically digital, tool support can provide additional benefits to efficiency, correctness and knowledge reuse, to name a few benefits [10].

As part of the Dependability Engineering Innovation for cyber-physical Systems (DEIS) research project[2], the concept of the Digital Dependability Identity (DDI) is being investigated [21]. DDIs are modular, composable and executable dependability information models associated with a CPS or its constituent subsystems or components. DDIs can be used as a medium for model-based security assessment and assurance, offering commensurate benefits to the development of security-critical CPS. The approach presented here will be employed in the context of a DDI.

[1] `https://ec.europa.eu/programmes/horizon2020/en/h2020-section/smart-cyber-physical-systems`

[2] `http://www.deis-project.eu`

In the following sections, a novel, model-based approach of systematically analyzing systemic security risks, identifying both high and low-level vulnerabilities and assigning appropriate requirements and measures will be presented. The approach will be demonstrated within the context of a CPS system for the healthcare domain. In section 2, previous work on security risk analysis will be reviewed. In section 3, our novel approach will be presented. Section 4 will describe the use case the approach is evaluated upon. Section 5 concludes by presenting the results, alongside further discussion of implications and future work.

2 Background

2.1 Security Threat And Risk Analysis

Due to the diverse applications for cyber-physical systems various industry-specific standards or best practices are applied. Each standard approaches risk differently depending on the factors deemed relevant to risk for the operating context. For example within the medical domain their is the IEC/TR 80001-2-1:2012, this is a technical report and guide on the application of risk management of medical IT networks it describes a 10 step process that system creators and maintainers can use in order to adhere to IEC 80001-1:2010 throughout a systems life-cycle.

Confidentiality is often a key requirement of any software system especially when dealing with sensitive personal data such as medical histories. In many countries personal data is covered by law, such as General Data Protection Regulation (GDPR) in European countries. Infringing on GDPR within the European Union can result in large fines of 4% of international annual turnover or €20 million depending on which is the greater (GDPR, Article 83).

The growing need to ensure privacy of data and the ever increasing capabilities and complexity of CPS has driven the development of frameworks and methodologies for privacy risk assessments and analysis's such as PRIAM (Privacy RIsk Analysis Methodology) which within the context of risk assessment breaks a system in to 7 components: the system itself including its logical boundaries, stake-holders, data, risk sources. privacy weaknesses, feared events and privacy harm [8]. Each component is comprised of categories and attributes. Categories describe the type of data of attribute for example this could be health data, financial data, location data etc and categories can be linked to other components. Attributes are used to identify the aspects of a component which contribute to privacy risk. They can be qualitative (low, medium, high) or quantitative such as "costs less than €5000". The application of PRIAM is divided in to two stages the information gathering stage where information on the components, categories and attributes is collated, and the risk assessment phase where risk levels (severity and likelihood) are calculated for each privacy item.

2.2 Security Attack Trees Analysis

Fault Tree Analysis (FTA) is an established practice in the domain of safety-critical applications. [3] showed that FTA can be applied in the domain of security-critical applications. Security Attack Trees (ATs) are similar to Fault Trees but specialised for the security domain. ATs provide a formal, hierarchical, model-based description of a system's security under a tree structure.

At the root of an AT are outcomes which represent security-critical negative events. Examples include maliciously gaining access to confidential information or obtaining administrator privileges for a safety-critical system. From the root node, intermediate nodes and logical gates link towards its leaf nodes. Intermediate nodes represent combined events that causally lead from their children to the root node. Logical gates usually represent Boolean logic operators such as AND and OR. A node's children linked via an AND gate describe that all of the events described in the children nodes are required for the parent node's event to occur. Accordingly, any event in a child's node is sufficient to trigger a parent node linked via an OR gate. Leaf nodes represent events which are out of scope or cannot be further analysed within the given AT. In the context of ATs, base events typically include direct, singular actions that form part of an attack.

An example of an AT can be seen in Fig.1, where a simplistic attack to gain administrator privileges on an abstract system is described. To achieve their goal, the attacker must either trick the system into executing privileged commands without authenticating as an administrator or successfully authenticate as an administrator (and then presumably execute any commands they wish). Each of these options are analyzed further; to execute commands without administrator authentication, the attacker can use a vulnerable user command and attach commands as the payload of a buffer overflow attack. Alternatively, to authenticate as an administrator, the attacker can use a brute force technique to discover the credentials or use a 'phishing' attack i.e. trick the administrator into disclosing them.

Each component of a system often has its own AT, these are combined to create the overall AT for a system. In this way, it is possible to reason about a complex system's security vulnerabilities in a modular fashion. This modularity is useful when dealing with system boundaries at different levels of abstraction.

Binary properties are often assigned to each node of an AT, such as Possible/Impossible, Expensive/Not Expensive, 'Special Equipment Required'/'No Special Equipment Required'. Numeric properties such as financial cost are also possible. Such properties can extend the AT, enabling quantitative systems security analysis. For example using the AT shown in 2, system creators can refine the AT by using the following query "attacks with an accumulative value of less than £5,000". This means the only attack that meets this criteria is threatening the legitimate administrator.

Such additional attributes can be described as 'resistance' attributes [22]. The example AT modified with such resistances can be seen in 2, where financial costs have been assigned to each leaf node. The semantics of the resistance attributes decide how they are combined as they move upwards through logic gates. In

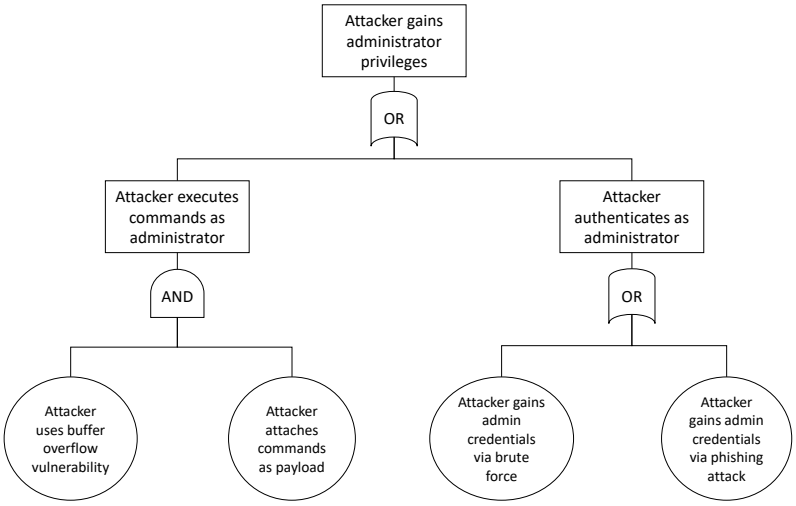

Fig. 1. Security Attack Tree Example

the example, the AND gate combines financial costs, as both actions must be undertaken, whereas the OR gate provides the attacker choice over the options; under the assumption that all the options are known and available, the attacker is also assumed to choose the most economic one available.

Buldas et al presented a Multi-Parameter Attack Tree, in this type of AT the assumption that an adversary will act within rationally and will not persevere with an attack if the cost outweighs the potential benefits [5]. This type of AT also means the relationship between attributes can be considered such as the overall effort involved and the competency of the adversary.

In fact an entire family of closely related AND-0R tree structures exists, which have been developed since ATs were first introduced, including, Attack-Defence Trees [13] and Ordered Weighted Average (OWA) Trees [23].

2.3 Security Capabilities and Controls

Security Capabilities and Controls are designed to protect systems against attacks on the confidentiality, integrity, and/or availability of a systems information. The USA's National Institute of Standards and Technology (NIST), defines a security control as a safeguard or countermeasure prescribed for an information system or an organization designed to protect the confidentiality, integrity, and availability of its information and to meet a set of defined security requirements. Additionally, NIST defines a Security Capability as a combination of mutually-reinforcing security controls (i.e., safeguards and countermeasures) implemented by technical means (i.e., functionality in hardware, software, and firmware), physical means (i.e., physical devices and protective measures), and procedural means (i.e., procedures performed by individuals)' [9].

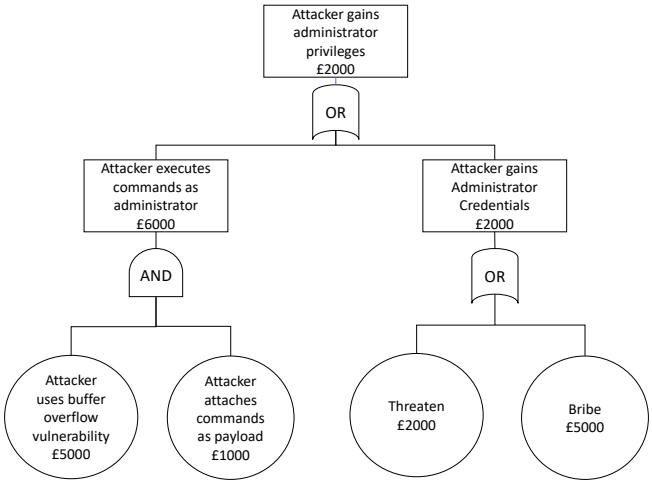

Fig. 2. Attack Tree with a Cost Resistance

There are a number of international standards and frameworks which promote good security practice in part by defining security capabilities and controls. The key considerations in choosing a framework include: understanding what an organisation needs to comply with from a contractual, statutory, and regulatory perspective; the comprehensiveness of the framework. Two of the most well-known frameworks include NIST SP 800-53 and the ISO 27000 series of standards which provide a framework for security management. While the fundamentals of both frameworks are largely the same, they differ in content and layout. Fig. 3 visualises the relationship between these two frameworks and indicates that ISO 27002 is a subset of NIST 800-53, as ISO 27002 has 14 security control categories which are encompassed by the 18 categories within NIST 800-53. Examples of such categories include: Incident Response; Access Control; and Audit and Accountability. NIST 800-53 is considered best practice within the US and vendors to the US government must meet its requirements. Outside the US, the ISO 27002 is the de-facto security framework and is considered less complex and easier to implement.

Another framework gaining in popularity is the NIST Cybersecurity Framework. It is more high level and concise than other frameworks and references NIST 800-53 and ISO 27002 for detail on how to implement specific controls and processes. As the NIST Cybersecurity Framework is more lightweight than the other existing frameworks, it may be more suitable for smaller organisations and more readable for executives who do not have a technical background.

More specific to the healthcare domain, which is the domain of the Use Case described in Section 4 of this paper, are the Health Information Trust Alliance (HITRUST) Common Security Framework (CSF) and the IEC 80001 series of

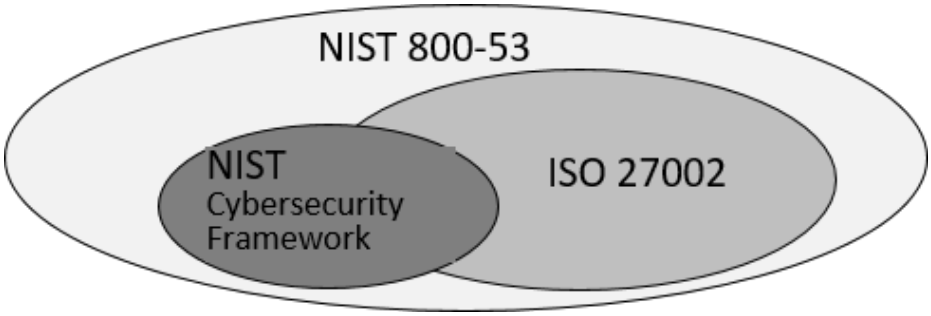

Fig. 3. Relationship between NIST 800-53 and ISO 27002

technical reports. The HITRUST framework incorporates healthcare-specific security, privacy and other regulatory requirements from existing standards such as ISO 27002 and is divided into 19 different domains or capabilities.The IEC 80001 series of technical reports provides guidance on the application of risk management for IT-networks incorporating medical devices. IEC TR 80001-2-8 provides guidance for the establishment of each of the security capabilities presented in IEC TR 80001-2-2 by identifying security controls from key security standards which aim to provide guidance to a responsible organisation when adapting the framework outlined in IEC TR 80001-2-2. IEC TR 80001-2-2 contains 19 security capabilities, with each capability having numerous security controls extracted from the following standards: NIST SP-800-53, ISO 27002, ISO/IEC 15408-2, ISO/IEC 15408-3, IEC 62443-3-3, ISO 27799. From these standards ISO 27002 and ISO 27799 are fully aligned. ISO IEC 27002 specifies a set of detailed controls for managing information security while ISO 27799 specifies additional guidance specifically for health information security and provides health information security best practice guidelines.

In Table 1, a small sample of security capability to controls mapping can be seen. Also included, are references to appropriate security standards, from where guidance on the controls can be referenced in detail.

Security Capability	Security Control	Reference
Transmission Integrity	Access Control for Transmission Medium	SP 800-53
	Network Controls	ISO IEC 27002...
	...	...

Table 1. Sample of Security Capability-Control Mapping

2.4 HiP-HOPS

Hierarchically Performed Hazard Origin and Propagation Studies (HiP-HOPS) is a well established method and tool in the field of dependability analysis [16]. HiP-HOPS has been successfully commercialised and adopted in industry[3]. It originally stems from the amalgamation of several classical dependability analysis techniques such as FTA and Failure Mode and Effects Analysis (FMEA). Its core function is the automation of such techniques with a view to increasing the quality (less mistakes) and the turn around time (efficiency) of dependability analysis of a system across its development lifecycle.

Over the past two decades, work has continued on HiP-HOPS and it has proven itself as a valuable lever for extending the corpus of research within dependable systems. For example, recently Papadopoulos et al showed that dependable systems design and analysis does not have to rely solely on advances in formal logic by using less conventional bio-inspired evolutionary techniques by extending HiP-HOPS to include meta-heuristics [17] and Kabir et al demonstrated that HiP-HOPS can be extended to create and analyse temporal fault trees using a systems architectural models [11] which can be used for constructing safety arguments.

Although HiP-HOPS has contributed to a steady stream of research and publications across the field of dependable systems analysis and design, the chief focus has been safety as an attribute of dependability. In terms of dependability it can be said the method presented here focuses on availability, confidentially and integrity, which are the composite properties that define security as an attribute of dependability [2].

Although alternative tooling for modelling ATs exists, such as SecurITree[4] and At-tackTree[5], they do not provide the required functionality and integration to automatically generate Digital Dependability Identities. This concept is explained in Section 2.5.

2.5 Digital Dependability Identity

The Digital Dependability Identity (DDI) [21] is a modular, composable and executable dependability model that links system structure with dependability information. The DDI offers numerous benefits. It enables convenient translation and exchange of heterogeneous dependability models across different tools and techniques. It supports execution of model-agnostic evaluation on its models, which in turn enables automation of assessment and assurance activities during design and monitoring and supervision during operation. As security is an aspect of dependability, the DDI also aims to capture security risk, threats, requirements, measures and other associated models. The Open Dependability

[3] `http://hip-hops.eu/`
[4] `https://www.amenaza.com/`
[5] `https://www.isograph.com/software/attacktree/`

Exchange (ODE) metamodel[6] is the DDI's metamodel and includes specific provisions for modeling security-related concepts. In particular, the ODE includes a TARA package, whose definition can be seen in Fig.4 .

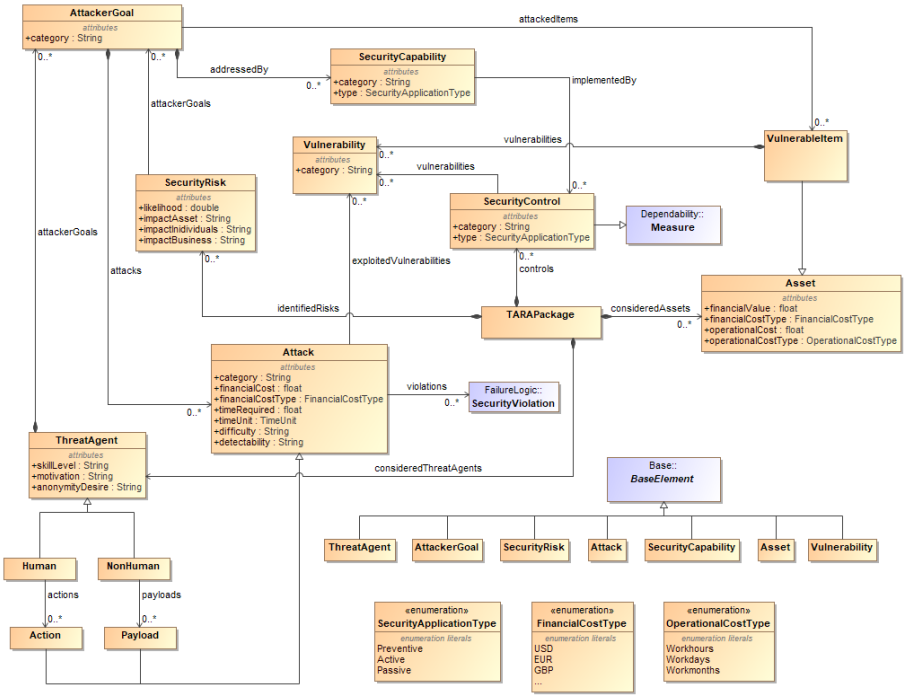

Fig. 4. ODE metamodel's TARA package definition

The package was defined drawing inspiration from the work in [15], as well as experience and deliberation across the DEIS project partners. The TARAPackage is the central containment unit, collecting security risks, measures, assets and 'Threat Agents' i.e. attackers under it. Risks are associated with the success of attacker goals, which are pursued via attacks, singular actions or payload executions. Attacks often exploit specific vulnerabilities that lie within the system's elements. Security capabilities and controls aim to address said vulnerabilities and safeguard the assets of a system.

Besides security aspects, the DDI can also encapsulate failure logic and reproduce fault trees, using an appropriate metamodel package. Analysis of DDI-

[6] `https://github.com/DEIS-Project-EU/DDI-Scripting-Tools/tree/master/ODE_Metamodel`

embedded fault trees and failure logic is supported by HiP-HOPS. More details can be found via the DEIS project's public deliverable D4.2[7].

3 Approach

We base our approach on a top-down life cycle development process, wherein system concept informs functionality. Dependability requirements (including security) are identified and allocated to functions. The above pattern repeats as functions decompose into more refined systems until low-level software/hardware components are specified. Following this lifecycle process, security capabilities are initially mapped to high-level security requirements. As requirements mirror the decomposition of systems to subsystems and components, low-level security requirements are addressed by the selection and, eventually, implementation of security controls.

To guide the choice of security capability and control selection, the TARA or equivalent risk analysis process can be applied to evaluate sources of security risk against the system. Threat agents, potential vulnerabilities, vulnerable system elements and other factors can be accounted for as part of the TARA. However, a TARA initially only addresses risk at a relatively high-level; further detail necessary to address the inner workings and complex relationships within the system architecture requires more refined techniques applicable both vertically (from systems to components) as well as horizontally (across the entire architecture). As per section 2.2, ATs are one such technique.

Following the TARA adopted by our approach, to launch a specific security attack, a threat agent must undertake a combination of actions or execution of payloads. The strategy of the attack is that security vulnerabilities in constituent system elements are exploited. The immediate effect of an action or payload execution can be viewed as a low-level event impacting security, referred to as a 'security violation'. Two examples of such a violation would be a malicious administrator introducing via portable storage and launching a malicious executable within an internal network. The attack in the example consists of an action - introduction of the executable by the malicious administrator - and a payload execution, each representing a security violation. Each security violation of the example can itself propagate and trigger further events. If security measures fail to address the chain of events, the attack is successful and the attacker's goal will eventually be reached, compromising one of the key assets that should have been protected.

Using ATs, the propagation logic that forms cause-effect chains of security violations to successful attacks can be described efficiently, with tool support providing all the benefits of model-based dependability analysis. What is required of the users is an appropriate annotation of the system architecture with local (i.e. per relevant system element) security violation propagation logic. Effectively, for

[7] http://www.deis-project.eu/fileadmin/user_upload/DEIS_D4.2_Engineering_Tools_for_DDIs_V1_PU.pdf

each element that can contribute or is affected by the propagation of security violations, the user should assign appropriate Boolean logic linking combinations of incoming or generated to outgoing security violations. Once this process is complete, automated security attack tree analysis tools, such as HiP-HOPS, can be used to perform qualitative analysis. The result of such an analysis is the identification of the necessary and sufficient combinations of security violations that can lead to attacks successfully compromising system assets.

Once the analysis results are available, the mapping provided in 2.3 can be used to plan appropriate security controls. For the scope of the work presented here, the mapping process will be limited to a simple look-up and selection from the list of available controls. In general, numerous criteria can be included in the decision process e.g. functional, design and financial constraints. The decision process itself can apply optimization strategies, both manual and semi-automatic, for further improvement.

4 ONCOAssist Use Case

ONCOAssist[8] is a mobile platform (available on IOS, Android and in a web format) for oncology professionals. It gives them a number of clinical tools and validated medical information that helps clinicians make a more informed decision when treating patients with cancer. As a use case, a clinician uses ONCOassist to calculate the body surface area and the drug dosage to be administered to the patient. It has been created by Portable Medical Technology Ltd, which are located in Ireland.

ONCOAssist interacts with private patient data and the users' account data as well. Since PMT is based in Ireland, they are subject to the GDPR as set out by the EU. Failure to comply with GDPR could result in the penalties mentioned in 2.1. The penalties of violating GDPR regulation due to ineffective security assessment, along with the expected security requirements healthcare establishments would likely set on their own before using ONCOAssist, necessitates that rigorous security assurances are provided.

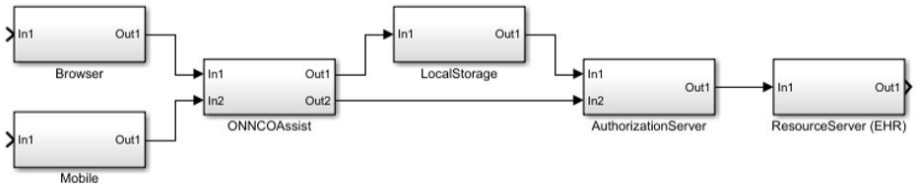

Fig. 5. ONCOAssist Authentication System Model

[8] `https://oncoassist.com/about/`

For brevity, authentication will be the focus of the case study, as the process can be illustrated using a relativity small system model. Before a user can perform any Create, Read, Update or Delete (CRUD) operations on the patient data stored in an Electronic Health Register (EHR), they must authenticate. Various open standards exist to be used as protocols for authentication. ONCOAssist uses OAuth2[9] with OpenID[10]. Using open standards such as these is often considered a good practice, as they have maximum exposure to public, third-party scrutiny.[1]

1. Establish a system model. Following discussions with and using activity diagrams provided by PMT we were able to produce a simplified abstract system model of ONCOAssist's authentication process using MATLAB, shown in figure 5. As seen in the figure, ONCOAssist can be accessed via a web application, with access to local device storage, authenticates with an authentication server, by requirement of the OAuth2 protocol, and finally retrieves patient data from a resource server, which is an EHR.

2. Establish a suitable TARA. Once we have established the system model, we can define feared events relevant to the system. In the case of ONCOAssist the feared events are data that becomes totally or partially compromised i.e. patient data that has been accessed to any extent by an unauthorised person.

3. Apply the TARA to the system model using HiP-HOPS. At this stage in the process, the system model is annotated with information gathered during the TARA using HiP-HOPS in conjunction with MATLAB's Simulink.

4. Conduct qualitative analysis on the AT produced by HiP-HOPS. Once the model is annotated, HiP-HOPS can be used to produce the SATs shown in Fig. 6.

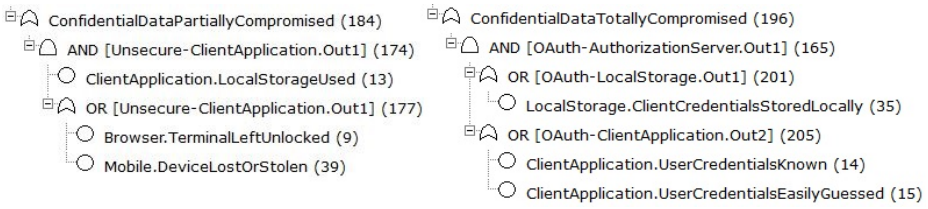

Fig. 6. ONCOAssist SATs generated by HiP-HOPS

5. Refine the system model as necessary and repeat 1-4. If necessary, further refinement can take place, depending on application, design and further considerations. For brevity, we limit our illustration to one iteration.

6. Capture the TARA, AT, HiP-HOPS results and system models within a DDI using the ODE. As mentioned in section 2.5, HiP-HOPS models can be exported as DDIs. Available tools developed by DEIS can convert embedded

[9] `https://oauth.net/2/`
[10] `https://openid.net/`

HiP-HOPS annotations into structures meaningful for DDIs such as the TARA and FailureLogic metamodel packages. Development of these tools is ongoing; an open-source version is available[11].

7. Use the Security Capability-Control mapping to validate requirements and identify controls for the DDI. Once the above information has been captured in a DDI, further tool support can be used to help requirements validation and appropriate control selection. Tool support developed by DEIS in this direction is provided in the form of executable scripts written in the Epsilon language[12]. Through Epsilon scripts, semi-automatic functionality can be designed and executed for DDIs generically. For instance, the process in Alg. 1 described in pseudocode can be implemented in the Epsilon Object Language (EOL)[12] for validating security capabilities and proposing controls.

```
foreach AttackerGoal g in subjectDDI do
    if g.addressedBy.size() = 0 then g is not addressed
    else
        foreach SecurityCapability sc in g.addressedBy do
            if sc.category != g.category then sc inappropriate for g
            else
                if sc.implementedBy.size() = 0 then sc is not implemented
                else
                    foreach SecurityControl c in sc.implementedBy do
                        if c.category != sc.category then c inappropriate for sc
                        else c implements sc
                    end
                end
            end
        end
    end
end
```

Algorithm 1: Security Capability Validation Diagnostic Example

5 Discussion and Future Work

Security analysis and development can be an expensive process; for example, the development of the ONCOAssist authentication system discussed in Section 4 represents approximately 6 weeks of work-hours for their development team, per PMT's account. However, failure to complete adequate security analysis can have catastrophic consequences. Therefore, efforts towards minimising this cost without compromising the quality of analysis are worthwhile. Digital

[11] `https://github.com/DEIS-Project-EU/DDI-Scripting-Tools/tree/master/ODE_Tooladapter`

[12] `https://www.eclipse.org/epsilon/`

management of SATs streamlines the process of security analysis by enabling automation (Security measures/controls and cost/benefit analysis can be derived semi-automatically) and by reducing the risk of human error.

Our proposed approach does have some notable limitations. Using HiP-HOPS for security analysis requires that system owners create and maintain an appropriately annotated system model. Thus, errors introduced in the model may critically compromise the subsequent analysis. For instance, such errors may occur due to lack of synchronization between the model and the implemented software, obscuring potential vulnerabilities present in the implementation. A further limitation of the process we have described is the lack of support for run time preventative security actions to be taken.

In summary, security and threat analysis is a complex process, often governed by regulation and frameworks have emerged to deal with this complexity. SATs are a valuable tool for security analysis and have been extended to deal with a wide variety of use cases. HiP-HOPS was originally created as a tool for traditional safety analysis but can be used for security analysis as well. DDIs are used to model a system's dependability, encompassing security concerns such as availability, confidentially and integrity. The approach described in section 3 and demonstrated in section 4 can be used for semi-automated security analysis. Considering future work, we are focusing on addressing run time security concerns. Specifically, appropriate methods for reasoning, negotiating and executing dependability-critical services using DDIs are the subject of ongoing research, as part of DEIS.

Acknowledgements

This work was supported by the DEIS H2020 Project under Grant 732242.

References

1. Adam Freeman, A.J.: Programming .Net Security. O'Reilly Media, 1 edn. (2003)
2. Avizienis, A., Laprie, J.C., Randell, B., Landwehr, C.: Basic concepts and taxonomy of dependable and secure computing. IEEE transactions on dependable and secure computing **1**(1), 11–33 (2004)
3. Brooke, P.J., Paige, R.F.: Fault trees for security system design and analysis. Computers & Security **22**(3), 256–264 (2003)
4. Bugliesi, M., Calzavara, S., Focardi, R.: Formal methods for web security. Journal of Logical and Algebraic Methods in Programming **87**, 110–126 (2017)
5. Buldas, A., Laud, P., Priisalu, J., Saarepera, M., Willemson, J.: Rational choice of security measures via multi-parameter attack trees. In: International Workshop on Critical Information Infrastructures Security. pp. 235–248. Springer (2006)
6. Cardenas, A.A., Amin, S., Sastry, S.: Secure control: Towards survivable cyber-physical systems. In: 2008 The 28th International Conference on Distributed Computing Systems Workshops. pp. 495–500. IEEE (2008)
7. Chong, S., Guttman, J., Datta, A., Myers, A., Pierce, B., Schaumont, P., Sherwood, T., Zeldovich, N.: Report on the nsf workshop on formal methods for security. arXiv preprint arXiv:1608.00678 (2016)

8. De, S.J., Le Métayer, D.: Priam: a privacy risk analysis methodology. In: Data Privacy Management and Security Assurance, pp. 221–229. Springer (2016)
9. Joint Task Force Transformation Initiative: Security and privacy controls for federal information systems and organizations. Tech. Rep. NIST SP 800-53r4, National Institute of Standards and Technology (2013). https://doi.org/10.6028/NIST.SP.800-53r4, https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf
10. Joshi, A., Miller, S., Whalen, M., Heimdahl, M.: A PROPOSAL FOR MODEL-BASED SAFETY ANALYSIS. In: 24th Digital Avionics Systems Conference. IEEE (2005). https://doi.org/10.1109/dasc.2005.1563469, https://doi.org/10.1109/dasc.2005.1563469
11. Kabir, S., Papadopoulos, Y., Walker, M., Parker, D., Aizpurua, J.I., Lampe, J., Rüde, E.: A model-based extension to hip-hops for dynamic fault propagation studies. In: International Symposium on Model-Based Safety and Assessment. pp. 163–178. Springer (2017)
12. Kolovos, D.S., Paige, R.F., Polack, F.A.: The epsilon object language (eol). In: European Conference on Model Driven Architecture-Foundations and Applications. pp. 128–142. Springer (2006)
13. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Foundations of attack–defense trees. In: Degano, P., Etalle, S., Guttman, J. (eds.) Formal Aspects of Security and Trust. pp. 80–95. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
14. Lee, E.A.: Cyber physical systems: Design challenges. In: 2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC). pp. 363–369. IEEE (2008)
15. Oates, R., Thom, F., Herries, G.: Security-aware, model-based systems engineering with sysml. In: Proceedings of the 1st International Symposium on ICS & SCADA Cyber Security Research. pp. 78–87. BCS (2013)
16. Papadopoulos, Y., McDermid, J.A.: Hierarchically performed hazard origin and propagation studies. In: International Conference on Computer Safety, Reliability, and Security. pp. 139–152. Springer (1999)
17. Papadopoulos, Y., Walker, M., Parker, D., Sharvia, S., Bottaci, L., Kabir, S., Azevedo, L., Sorokos, I.: A synthesis of logic and bio-inspired techniques in the design of dependable systems. Annual Reviews in Control **41**, 170–182 (2016)
18. Pasqualetti, F., Dörfler, F., Bullo, F.: Attack detection and identification in cyber-physical systems. IEEE transactions on automatic control **58**(11), 2715–2729 (2013)
19. Rajkumar, R., Lee, I., Sha, L., Stankovic, J.: Cyber-physical systems: the next computing revolution. In: Design Automation Conference. pp. 731–736. IEEE (2010)
20. Rivera, J.: Cyber security via formal methods: A framework for implementing formal methods. In: 2017 International Conference on Cyber Conflict (CyCon US). pp. 76–81. IEEE (2017)
21. Schneider, D., Trapp, M., Papadopoulos, Y., Armengaud, E., Zeller, M., Höfig, K.: Wap: digital dependability identities. In: 2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE). pp. 324–329. IEEE (2015)
22. Whitley, J.N., Phan, R.C.W., Wang, J., Parish, D.J.: Attribution of attack trees. Computers & Electrical Engineering **37**(4), 624–628 (2011)
23. Yager, R.R.: Owa trees and their role in security modeling using attack trees. Information Sciences **176**(20), 2933 – 2959 (2006). https://doi.org/https://doi.org/10.1016/j.ins.2005.08.004, http://www.sciencedirect.com/science/article/pii/S0020025505002598