COMPOSITE RISK MODELING FOR AUTOMATED THREAT MITIGATION IN MEDICAL DEVICES

Aakarsh Rao
Dept. of Electrical and Computer Engineering
University of Arizona
Tucson, AZ, USA
aakarshrao7@ece.arizona.edu

Jerzy Rozenblit
Dept. of Electrical and Computer Engineering
University of Arizona
Tucson, AZ, USA
jr@ece.arizona.edu

Roman Lysecky
Dept. of Electrical and Computer Engineering
University of Arizona
Tucson, AZ, USA
rlysecky@ece.arizona.edu

Johannes Sametinger
Dept. of Information Systems - Software Engineering
Johannes Kepler University
Linz, Austria
johannes.sametinger@jku.at

ABSTRACT

Medical device security is a growing concern with increasing incorporation of complex software and hardware. Security threats exploiting vulnerabilities in medical devices may directly impact patient safety. Standardization and federal organizations are hence, actively involved in setting up new paradigms for guidance and regulation of security throughout the lifecycle. To protect medical devices against threats a risk-based framework that continually manages and assesses security risks along with their proactive addressing is highly recommended. In this paper, we model a multi-modal design approach for risk assessment in medical devices and propose an adaptive remediation scheme to mitigate security threats. Our multi-modal approach is integrated into the hardware-software development with a middleware for interaction between the modes. This provides an effective premarket risk management while the adaptive remediation scheme pro-actively mitigate risk during postmarket deployment. We model our approaches in detail and demonstrate them in a pacemaker design model and deployment scenario.

Keywords: medical device security, security threats, risk management, hardware-software design.

1 INTRODUCTION

With advances in technology, medical devices are not far behind, consisting of composite embedded systems with expansive software and hardware elements. This has resulted in increased proliferation and notable improvements of quality of care and convenience for patients (Neuman et al. 2012). Furthermore, medical devices in recent times have been equipped with connectivity and interoperability to improve the alarming function, remote device support and autonomous control (Goldman 2006).

Security in medical devices, particularly implantable medical devices (IMD) is a special case of embedded systems and imposes special challenges (Sametinger et al. 2015). Risk management and assessment has emerged as a consensus standard and guidance for premarket and postmarket management of security in

SpringSim-MSM 2017, April 23-26, Virginia Beach, VA, USA

medical devices containing software-hardware (FDA 2005) (FDA). It is also required for manufacturers to identify hazards, analyze associated risks, control these risks and evaluate these controls (ISO 14971 2007). Risk management has to be considered throughout the product lifecycle from design to deployment. During the premarket phase, it is required that manufactures submit a comprehensive risk management report for the containing software and hardware components. To quantify this, the FDA also requires the inclusion of "Level of Concern" with the associated risks, that estimates the severity of patient safety *vis-a-vis* compromise of these components. In order to proactively and robustly manage security vulnerabilities, it is strongly recommended to consider risk management and assessment even during postmarket deployment of medical devices along with implementations of corresponding remediations to mitigate these risks. A key challenge to be considered during postmarket risk management and mitigation is to maintain safety and essential functioning of the device even if it has been compromised/exploited.

Several attacks have been successfully conducted exploiting vulnerabilities in medical devices. One of the first attacks on medical devices was demonstrated on pacemakers and implantable cardiac defibrillators (Halperin et al. 2008), while (Li, Raghunathan, and Jha 2011) hijacked an insulin pump. Both these attacks exploited the wireless communication channel and eavesdropped the cleartext information from the communication. With such possibilities of threats exploiting vulnerabilities in medical devices, a probable countermeasure would be to isolate medical device hardware and software functionality into different modes and switch between them based on the risk and threat involved (Sametinger and Rozenblit 2016). Such fail-safe modes should ensure device's essential functionality even if exploited. We explore this direction and suggest in this paper: 1) a multi-modal model for software-hardware design of medical devices specifically IMDs based on isolation of functionality with risk evaluation and 2) middleware incorporating an adaptive mitigation scheme to continually protect the device against exploits and vulnerabilities that may arise during postmarket deployment without hampering critical functionality. We provide details of our models and demonstrate our proposed design and adaptive mitigation scheme in presence of an exploit/vulnerability in a pacemaker scenario.

In the rest of the paper, we describe details of our proposed multi-modal design approach along with the middleware mitigation scheme in Section 2, demonstrate the same in a pacemaker model use-case scenario in Section 3 and conclude with future work.

2 PROPOSED FRAMEWORK MODEL

An illustration of the overarching working model incorporating our risk model framework along with the mitigation scheme via middleware in a medical device is shown in Figure 1. It is shown to illustrate our approach for embedding security mitigation mechanisms into medical device development.

2.1 Multi-modal Device Design

It has been well established in research and regulatory bodies that security has to be considered from the design stage of embedded systems (Kocher et al. 2004). Security has been augmented with risk evaluation and management especially in safety critical embedded systems like medical devices. The main aspects to be considered during design of medical devices are: 1) maintain essential functionality even if the device is exploited, 2) risk levels can be assigned and evaluated to be utilized for mitigation and 3) does not hamper the performance and resource requirements to a considerable extent.

We discuss the modeling of the risk model required for the design. The risk model is used to assign risk values to each function in every mode. The risk model is modeled by the designer in consent with a doctor specific to the medical device. The risk values assigned are an aggregation of safety and security concerns (Sametinger and Rozenblit 2016). A fine-grained assignment of risk values is required to ensure

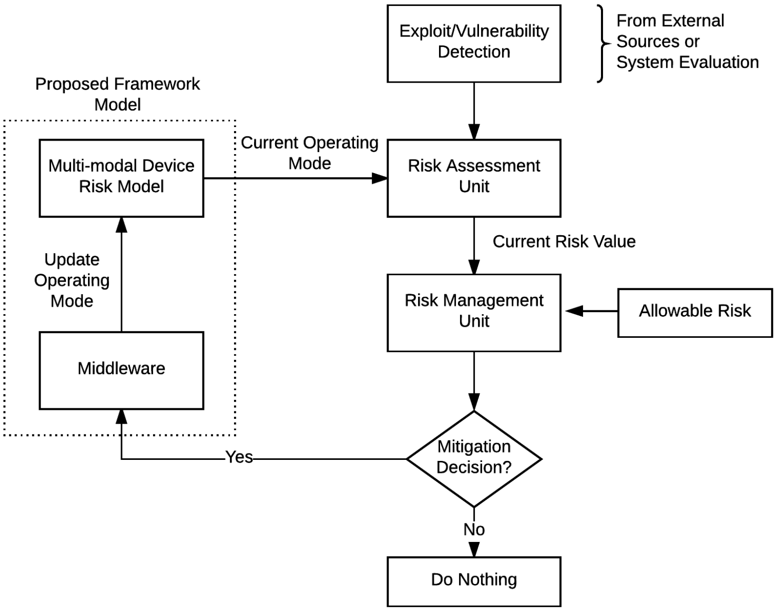

Figure 1: Overall Working Model.

robust risk management. Risk values for every mode are cumulated to give a *cumulative risk value* that represents the overall risk of operating in that specific mode. The *cumulative risk value* is utilized by the mitigation scheme to determine if there is a need for mitigation based on an allowable risk for each operating mode.

We suggest a new design paradigm of hardware and software design for medical devices that considers these aspects. This paradigm is based on *isolating functionality* of the medical device and *evaluating corresponding risks values*. Functionality or application can refer to either a piece of software performing on-device actions or part of the firmware/software controlling a specific hardware component. The functionality or applications running on a medical device are required to be isolated to different modes rather than the traditional design of "chunk" of code running on hardware. To incorporate security deeper into the design, isolation of fine-grained functionalities is highly beneficial. Fine-grained functionalities will include access to a hardware component, data transfer, control signals, software classes and its members. However, it is challenging to abstract medical devices, so our design methodology has to be tailored to a specific device based on it's functionalities.

Here, we explore the specifics of the modes. The *base mode* or *Mode 0* represents the bare-bones functionality of the device. In other words, this mode of operation represents the critical functions of the device required for essential functioning. This is the safest mode of operation for the device if a vulnerability or threat has been detected. We envision the functionalities in this mode to be secure by having safeguards for software and corresponding hardware. This can be implemented using secure technology like the ARM TrustZone ® (ARM 2009). For example, in the case of an IMD, *Mode 0* would contain functions needed to keep the device working or in other words, keep the patient alive. Additional functionalities are abstracted to separate modes *Mode 0, Mode 1, ...* in a decreasing order of functional criticality. The constraint to design higher modes would be a monotonically increasing cumulative risk value. We correlate the functional criticality to security and safety because, if there was a potential of exploit or a vulnerability in a non-critical function of the device, that threat would not penetrate down to the essential functions of the device as they have been isolated to a lower mode.

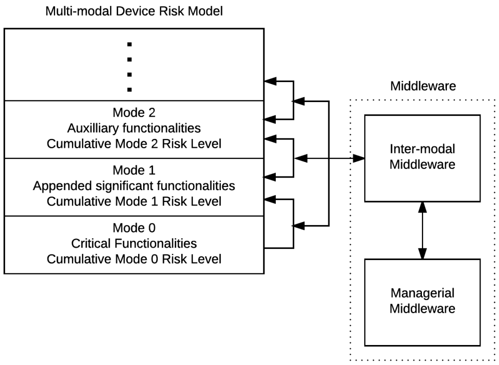
Multi-modal Device Risk Model
Mode 2
Auxilliary functionalities
Cumulative Mode 2 Risk Level
Mode 1
Appended significant functionalities
Cumulative Mode 1 Risk Level
Mode 0
Critical Functionalities
Cumulative Mode 0 Risk Level
Middleware
Inter-modal Middleware
Managerial Middleware

Figure 2: Detailed Risk Model with Middleware.

Invariably, there will be a need to transfer data and/or signals bi-directionally between the modes. This interface should not expose the multi-modal design to security risks; for which, we suggest an *inter-modal middleware*. The *inter-modal middleware* is responsible for the secure transfer of data and signals between the respective modes as well as mitigation.The *inter-modal middleware* will have to be implemented with secure technology (ARM 2009) with secure pointers/methods to the respective mode functionalities for the transfer.

2.2 Adaptive Mitigation Scheme

For risk management in real-time during postmarket deployment, we propose an adaptive mitigation scheme that takes as inputs the cumulative risk value of the current operating mode and compares it with a pre-set configurable allowable risk value to determine if mitigation is necessary. The allowable risk value can be set by the designer in consent with a expert and is configurable depending on updates. Since we have isolated the functionalities to different modes, vulnerabilities or exploits are also restricted to specific modes, which shows the benefit of our design methodology. We assume that the vulnerabilities or exploits can be detected in a device as demonstrated by the works (Lu, Seo, and Lysecky 2015) (Maxion and Tan 2002) (Qin et al. 2006). The middleware is responsible for the mitigation scheme and composed of: 1) *inter-modal middleware* for vulnerable component cut-off (in addition to communication between modes as in Subsection 2.1) and 2) *managerial middleware* for switching between operating modes, see Fig. 2.

Specifically, a *managerial middleware* is presented to handle the mitigation strategy. We assume the *managerial middleware* to be implemented using secure technology (ARM 2009). It contains complete information of the modes of operation, risk model and current mode of operation of the device assisted by its connection to the *inter-modal middleware*. We show in Algorithm 1, the implementation procedure of the middleware with the detailed description of the actions as follows:

1. The risk value of the functions increases with the presence of a vulnerability or exploit as it is responsible for requesting data/signals or accessing hardware components that the *inter-modal middleware* detects is not a part of the inherent mode. The rate of increase of risk value depends on the assigned risk model for that particular data/signal or method or in other terms, the criticality of the functionality. If the updated evaluated cumulative risk value is greater than the allowable risk then the mitigation is triggered, otherwise the device works as normal.

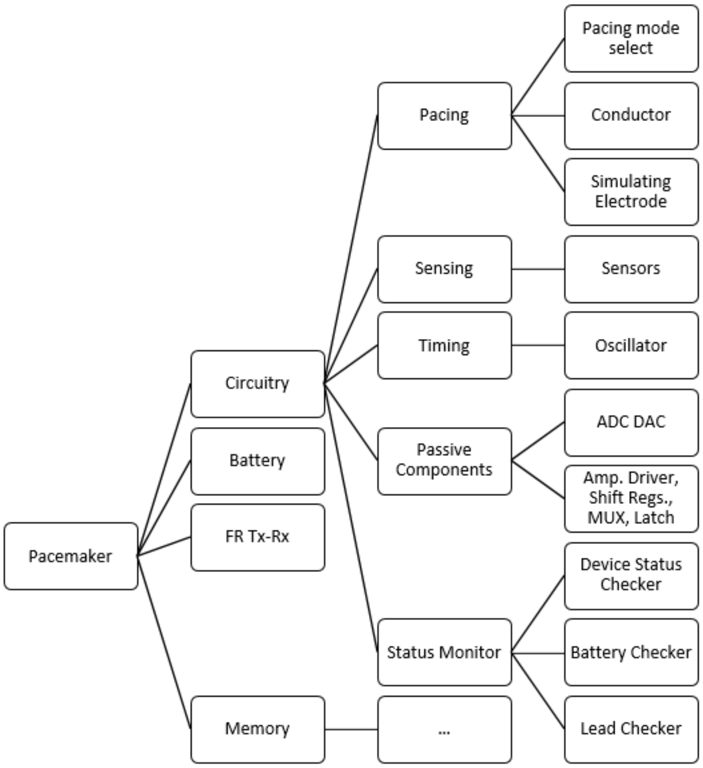

Figure 3: Hierarchical components of a rate-adaptive demand pacemaker.

2. Upon mitigation, the *inter-modal middleware* forbids access to the requested data/signal or method by the vulnerability or exploit. It will also cut-off the component or method affected by this vulnerability or exploit thus effectively reducing the cumulative risk value. This request from an unspecified source is logged, similar to an exception type mechanism.
3. We envision the presence of a *device risk level* as well. This represents the value at which the device is unsafe to operate at the current mode of operation due to severe security threats to critical functionalities. If the evaluated risk has risen beyond the *device risk level*, the *managerial middleware* will be informed by the *inter-modal middleware*. This will effectuate a shift to a lower mode of operation, thus mitigating the threats by forbidding the entire mode. A fail-safe method of mitigation can also be implemented that shifts the device directly to the *Mode 0* or *base mode* of operation, which ensures the essential functioning of the device as well as mitigation of threats by reducing the cumulative risk value.
4. If the vulnerabilities or threats have been addressed, then the device can be reset to a higher mode of operation.

3 USE-CASE SCENARIO: PACEMAKER

We provide a use-case scenario of a pacemaker, to show how our methodology can adaptively mitigate threats by utilizing a multi-modal approach using a risk model. We choose the pacemaker since it is a widely popular Implantable Medical Device (IMD). A pacemaker is a small device that regulates the beating of the heart in order to treat heart conditions. Modern pacemakers are complex medical devices, and there are several types of pacemakers to treat various conditions with a wide range of functionalities (Medtronic 2016). To simplify the demonstration for the scope of this paper, we model a simple rate-adaptive demand

Algorithm 1 Middleware Implementation Pseudo-code.

```
Input: Risk value of methods $rv_{method}$, Allowable risk at mode $arv_{mode}$, Cumulative risk vale $crv_{mode}$,
    Device risk value $drv$, Current mode of operation $mode_{curr}$
Output: Component/Method cut-off signal $S_{comp/method}$, Mode shift()
procedure MIDDLEWARE($rv_{method}, arv_{mode}, mode_{curr}, crv_{mode}, drv$)
    if Vulnerability detected then
        Get $method_i$ affected
        Update $rv_{method_i}$ and Evaluate $crv_{mode}$
        Send $S_{method_i}$
        if $crv_{mode_{curr}} > drv$ then
            Mode Shift ($mode_{curr}$)
        end if
    end if
end procedure
```

pacemaker as our use-case, based on Medtronic Inc. (Bornzin 1984). A rate-adaptive demand pacemaker is a pacemaker that generates a pulse only when a natural one is missing. In addition, a physiological parameter is utilized to determine the rate of pulse that needs to stimulated. The hierarchy of components of our pacemaker is illustrated in Figure 3 to aid in the risk modeling.

3.1 Rate-adaptive Demand Pacemaker Risk Model

We obtain the functionalities for the rate-adaptive demand pacemaker from (Bornzin 1984). The risk model in a tabular fashion is shown in Table 1. The components are bifurcated to their corresponding attributes. For each attribute, we have actions that are associated with them and at this "action granularity" we assign the risk values. The purpose of this design is to ease in abstracting a pacemaker with additions only required for attributes and actions. Our risk values range from 1 - 10, where 1 represents least risk (every component will inherently have some risk) and 10 the highest risk. The risk values within a specific attribute is relatively assigned. We describe an example to comprehend the risk model.

Pacing is the component of the pacemaker responsible for calculating the pacing interval to be stimulated to the leads that are attached to the heart. To provide the prescribed therapy, an attribute of this component is the *Compute Therapy Parameters*. The key actions attributed are to *compute escape interval(), pacing enable() and pacing rate control()*. The *compute escape interval()* calculates the escape interval for the pacemaker based on the oxygen sensor. We assign a risk value of 6 as changes to this calculation will pose a higher threat to the critical functioning of the pacemaker. On the other hand, *pacing enable()* is assigned a higher risk of 8 since it is relatively more critical than *compute escape interval()* as this action decides if a natural pulse has occurred based on the EKG electrode signal and escape interval, that enables/disables a pacemaker induced pulse to the heart.

3.2 Multi-modal Pacemaker Design

The designer is free to choose as many modes as required based on the security granularity required. For the sake of illustration, we build two modes for the specified rate-adaptive demand pacemaker. As stated in Subsection 2.1, the *Mode 0* (Table 2) represents the essential functionalities, which in a pacemakers case refers to the continual stimulation of electrical pulses to the heart. This would imply: 1) a real-time detection of the oxygen concentration in the blood, 2) computing the pulse rate proportional to the concentration, 3)

Table 1: Pacemaker Risk Model.

Component	Attributes	Actions	Base Risk Value
Sensors	Oxygen Sensor	Read sensor value()	1
	EKG Electrode		5
Pacing	Stimulate Lead	Compute pace voltage()	8
		Send lead voltage()	10
	Pacing Mode Select	Pace mode select()	8
	Compute Therapy Parameters	Compute escape interval()	7
		Pacing enable()	8
		Pacing rate control()	7
Timing	Oscillator	Compute oscillator period()	9
Status Monitor	Battery Checker	Read battery value()	1
		Write battery value()	4
		Wireless transfer()	7
	Lead Checker	Read lead impedance()	1
		Write lead impedance()	7
		Wireless transfer()	7
Memory	Programmable parameters	Wireless transfer()	1
		Write prog. parameters()	8
	Fixed Parameters	Read fixed parameters()	4

computing the pace voltage level from the pulse period, 4) supplying the regulated voltage to the set leads and 5) the leads are set based on the pacing mode pre-set by the physician based on the heart condition and oxygen concentration. The respective risk values are assigned to the actions in *Mode 0* and in certain actions an average risk computed that would represent the attribute. The cumulative risk value is calculated as simply the sum of the assigned risk values. Details of *Mode 1* are self explanatory and represented in Table 3. Furthermore, *Mode 1* should always have a higher *cumulative risk value* compared to *Mode 0* (35 for *Mode 0* and 37 for *Mode 1* in our case). The inter-modal middleware (Subsection 2.2) is responsible for secure interaction between *Mode 0* and *Mode 1*; here to transfer the oxygen concentration value and pacing enable signal.

3.3 Demonstration of Adaptive Mitigation Scheme

We demonstrate the working of the adaptive mitigation scheme under two cases: 1) vulnerability existing in one of the actions and 2) presence of on-device malware. It is a fair assumption that by default the device would operate in the higher mode i.e. *Mode 1* in our pacemaker model.

3.3.1 Vulnerability in Actions

If a vulnerability is detected in an action module, the *inter-modal middleware* increments the basic risk value associated with it based on the input from the vulnerability detection unit. The incremental model will be explored in future work, but we envision it depending on the influencers and influencees of the action. This change in risk value will lead to the *inter-modal middleware* temporarily cutting off that component from the current mode of operation. Also, the evaluated *cumulative risk value* is sent to the *managerial middleware* to check if a switch of mode is required. If the vulnerability is addressed, the risk value would be reset and component re-activated. For example, if the *lead checker()* action develops a vulnerability that causes

Table 2: Mode 0 Model.

Mode 0 Actions	Risk values
Read sensor value(Oxygen)	1
Read fixed parameters from ROM Read_fixed_parameters()	4
Algorithm to select appropriate pacing mode based on diagnosis from sensors and parameters Pacing_mode_select( )	8
Algorithm to compute required pacing period from oscillator Compute_oscillator_period( )	3
Algorithm to convert pacing period to appropriate voltage to be supplied to the leads Computer_pace_voltage( )	8
Convert battery supplied voltage to the pacing voltage using passive electronics	1
Pacing voltage transferred to actuator/conductor to stimulate leads Send_electrode_voltage( )	10
Cumulative Risk of Mode 0	35

erroneous changes to the lead impedance values, the *inter-modal middleware* increments the risk value from 5 to say 7 and cuts of this action (*lead_checker()*) from operating through the *cutoff(lead_checker())* method, thus mitigating the vulnerability till it's addressed.

3.3.2 On-device Malware

An on-device malware disrupts essential functionalities of the pacemaker by manipulating the pulse rate that has to be stimulated to the heart. This will require requests to *inter-model middleware* or will have to manipulate actions that are not allowed in the current mode of operation. The *inter-modal middleware* will detect the absence of the malware method in it's specification and notify the *managerial middleware* of this new malware method along with the action/data its trying to manipulate i.e. *pulse rate* and *compute_oscillator_period()*. The *managerial middleware* detects the threat to affect the critical functioning of the device due to which the *cumulative risk level* of *Mode 1* goes beyond the *device risk level*. This results the *managerial middleware* immediately shifts the mode of operation of the pacemaker from *Mode 1* to *Mode 0* via *shift_mode()* method. This effectively assuring the continual essential functioning of the device as well as mitigating the malware.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new design paradigm for critical embedded devices especially medical devices to provide security as well as ensure safety during both premarket and postmarket deployment. The design approach is a multi-modal one that allows the designer to abstract hardware components and their corresponding software methods to different modes based on their criticality. To assist in risk evaluation and management, the design is associated with a risk model and is used to assign risk values to the components. An adaptive mitigation scheme utilizing a *managerial middleware* actively mitigates exploits or vulnerabilities by cutting of components in a specific mode or shifting operating modes upon evaluation of the risk value associated. We demonstrate this design and mitigation scheme in a rate-adaptive demand pacemaker use-case. The model framework proposed with the mitigate scheme seems promising. In future

Table 3: Mode 1 Model.

Mode 1 Actions	Risk values
Read sensor value(Natural Physiology sensor)	5
Algorithm to calculate escape interval based on current oxygen sensor reading Compute_escape_interval()	7
Algorithm to check if pacing is required based on natural physiology sensor Pacing_enable()	8
Program to Read-Write lead impedance value – lead checker Read_lead_impedance( ), Write_lead_impedance( )	5
Program to Read-Write battery value – battery checker Read_battery_value( ), Write_battery_value( )	4
Write programmable parameters input by doctor through monitoring device Write_prog_parameters( )	8
Cumulative Risk of Mode 1	37

work, we anticipate pursuing a formal proof of the working model and a certification of our method with a hardware-software implementation.

REFERENCES

ARM 2005-2009. *ARM Security Technology*. ARM Limited. Available http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf. Accessed Jan. 2, 2017.

Bornzin, G.A. 1984, August 28. "Rate adaptive demand pacemaker". US Patent 4,467,807.

FDA. *Postmarket Management of Cybersecurity in Medical Devices*.

FDA 2005. *Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices*. FDA - U.S. Food and Drug Administration. Available http://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/ucm089593.pdf. Accessed Jan. 12, 2017.

Goldman, J. M. 2006. "Medical Device Connectivity for Improving Safety and Efficiency". *ASA Monitor* vol. 70 (5), pp. 6–7.

Halperin, D., T. S. Heydt-Benjamin, B. Ransford, S. S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. H. Maisel. 2008. "Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses". In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pp. 129–142. IEEE.

ISO 14971 2007. *Medical devices - Application of risk management to medical devices*. ISO. Available http://www.iso.org/iso/catalogue_detail?csnumber=38193. Accessed Jan. 2, 2017.

Kocher, P., R. Lee, G. McGraw, and A. Raghunathan. 2004. "Security As a New Dimension in Embedded System Design". In *Proceedings of the 41st Annual Design Automation Conference*, DAC '04, pp. 753–760. New York, NY, USA, ACM. Moderator-Ravi, Srivaths.

Li, C., A. Raghunathan, and N. K. Jha. 2011, June. "Hijacking an insulin pump: Security attacks and defenses for a diabetes therapy system". In *2011 IEEE 13th International Conference on e-Health Networking, Applications and Services*, pp. 150–156.

Lu, S., M. Seo, and R. Lysecky. 2015, Jan. "Timing-based anomaly detection in embedded systems". In *The 20th Asia and South Pacific Design Automation Conference*, pp. 809–814.

Maxion, R. A., and K. M. C. Tan. 2002, Feb. “Anomaly detection in embedded systems”. *IEEE Transactions on Computers* vol. 51 (2), pp. 108–120.

Medtronic 2016. *Pacemakers*. Medtronic. Available http://www.medtronic.com/us-en/healthcare-professionals/products/cardiac-rhythm/pacemakers.html. Accessed Jan. 4, 2017.

Neuman, M. R., G. D. Baura, S. Meldrum, O. Soykan, M. E. Valentinuzzi, R. S. Leder, S. Micera, and Y. T. Zhang. 2012, May. “Advances in Medical Devices and Medical Electronics”. *Proceedings of the IEEE* vol. 100 (Special Centennial Issue), pp. 1537–1550.

Qin, F., C. Wang, Z. Li, H. s. Kim, Y. Zhou, and Y. Wu. 2006, Dec. “LIFT: A Low-Overhead Practical Information Flow Tracking System for Detecting Security Attacks”. In *2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO’06)*, pp. 135–148.

Sametinger, J., and J. Rozenblit. 2016. *Security scores for medical devices*, pp. 533–541. SciTePress.

Sametinger, J., J. Rozenblit, R. Lysecky, and P. Ott. 2015, March. “Security Challenges for Medical Devices”. *Commun. ACM* vol. 58 (4), pp. 74–82.

AUTHOR BIOGRAPHIES

AAKARSH RAO is currently a PhD student in the Electrical and Computer Engineering Department at the University of Arizona. He works in the Model Based Design Laboratory on medical device security and design. He received his Masters in January 2015 from the same university and B.S in Electronics and Communication from M.S. Ramaiah Institute of Technology, India in 2009. His email is aakarshrao7@email.arizona.edu.

JERZY ROZENBLIT is a University Distinguished Professor, Raymond J. Oglethorpe Endowed Chair in the Electrical and Computer Engineering (ECE) Department, with a joint appointment as Professor of Surgery in the College of Medicine at the University of Arizona. During his tenure at the University of Arizona, he established the Model-Based Design Laboratory with major projects in design and analysis of complex, computer-based systems, hardware/software codesign, and simulation modeling. He presently serves as Director of the Life-Critical Computing Systems Initiative, a research enterprise intended to improve the reliability and safety of technology in healthcare and life-critical applications. His email address is jr@ece.arizona.edu.

ROMAN LYSECKY is an Associate Professor of Electrical and Computer Engineering at the University of Arizona. He received his B.S., M.S., and Ph.D. in Computer Science from the University of California, Riverside in 1999, 2000, and 2005, respectively. His research interests focus on embedded systems, with emphasis on runtime optimization, non-intrusive system observation methods for in-situ analysis of complex hardware and software behavior, data-adaptable system, and embedded system security. He received the Outstanding Ph.D. Dissertation Award from the European Design and Automation Association (EDAA) in 2006, a CAREER award from the National Science Foundation in 2009, and five Best Paper Awards. His email is rlysecky@ece.arizona.edu.

JOHANNES SAMETINGER is an Associate Professor in the Department of Information Systems at the Johannes Kepler University Linz, Austria. He holds a Dr. techn. in computer science from the Johannes Kepler University Linz. His research interests include software engineering and IT security with an emphasis on software security and medical device security. He has made several research visits to universities in the United States and in Canada, including the University of Arizona. His email address is johannes.sametinger@jku.at.