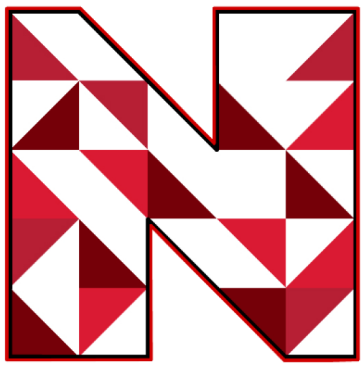

NEBRASKA APPLIED RESEARCH INSTITUTE

at the University of Nebraska

On Securing Life-Critical Systems

How the science of securing life-critical systems can be advanced

Report Info	
Authoring Organization:	The Nebraska Applied Research Institute (NARI)
Supporting Organizations on NARI Team	The University of Nebraska Medical Center (UNMC) Departments of Cardiology and Endocrinology The University of Nebraska, at Omaha (UNO) College of Information Science and Technology (IS&T) Diabetes Technology Society
Type of Organization:	501(c)(3) Non-Profit
Website:	https://nari-cyber.com
Report Date:	3/31/2020
Revision	v 2.0
Authors	Owen Redwood, Ph.D., NARI Robin Gandhi, Ph.D., UNO IS&T Matt Hale, Ph.D., UNO IS&T Cyrus Desouza M.D., UNMC Endocrinology Andjela Drincic M.D., UNMC Endocrinology Leslie Eiland M.D., UNMC Endocrinology Faris Khan M.D., UNMC Cardiology William J. Schleifer M.D., UNMC Cardiology David C. Klonoff, M.D., President of Diabetes Technology Society Eric Wright, NARI Ian Trent, NARI Riley Hester, NARI Heather Lawrence, NARI Jeff Dunn, NARI Orly Tauil, NARI Brad Miller, NARI
Technical Point of Contact:	W. Owen Redwood, Ph.D. Interim Executive Director & Chief Research Officer 6825 Pine Street, Ste. 150, Omaha, NE 68106 Tel: 402.554.6297 / E-mail: Owen.Redwood@nari.nebraska.edu
Administrative Point of Contact:	Jen Hale Business Operations Specialist 6825 Pine Street, Ste. 150, Omaha, NE 68106 Tel: 402.554.6274 / E-mail: Jen.Hale@nari.nebraska.edu

THIS RESEARCH WAS DEVELOPED WITH FUNDING FROM THE DEFENSE ADVANCED RESEARCH PROJECTS AGENCY (DARPA). This report was produced in fulfillment of the Patching Life Critical Systems subcontract number PO-0017642 between the Nebraska Applied Research Institute and Perspecta Labs, under U.S. Government Prime Contract Number FA8750-16-C-0178.

Abstract

Cyberattacks against healthcare critical infrastructure are sharply on the rise and are trending to outpace the defensive advancements of the collective healthcare sector. These threats are likely to lead to significant impacts to public health and safety, national security, economic security, and civil liberties. To advise the community on how to address these significant threats, Nebraska Applied Research Institute (NARI) supported by practicing medical researchers at the University of Nebraska Medical Center (UNMC), computer science professors at the University of Nebraska at Omaha (UNO), and the president of the Diabetes Technology Society (DTS) have performed a wide spectrum of exploratory research on securing life-critical biomedical devices. This study demonstrates how the science of patching and securing biomedical devices could be improved and even industrialized to the levels currently enjoyed by other critical infrastructure sectors.

Biomedical device cybersecurity hygiene is often non-existent in practice, with systems going unpatched, default passwords going unchanged, and clinical staff going unaware of the basics of cybersecurity hygiene. Given this lack of cybersecurity hygiene awareness we liken the state of biomedical cybersecurity to Medieval clinical hygiene. Patching systems is as important to cybersecurity as hand-washing is to avoid infection. Unfortunately securing and patching systems is not as straightforward as hand-washing. This study presents the challenges and potential solutions to securing and patching biomedical systems considering perspectives of key stakeholders including healthcare delivery organizations, patients, and vendors.

This study explores challenges and potential solutions across the entire lifespan of these devices from design-phase to legacy-phase. Our team analyzed the processor and firmware details for the top life-critical devices on the market today so that we can best inform future efforts with the most accurate information. To assess significant challenges in this space, we performed exploratory research and development, and produced several proof of concepts and findings. Additionally, we conducted medical community surveys of doctors, patients, and medical device manufacturers capturing the scale and significance of the cybersecurity and patching challenges which demonstrates the importance, relevance, and timeliness of the findings. To advance the science of securing life-critical systems our feasibility study synthesizes findings to offer concrete evidence-based recommendations. They include additional research and development in the following key areas with revolutionary potential, high risk, high reward opportunities, such as:

- [redacted].
- [redacted].
- [redacted].
- New virtualization capabilities for software assessment.

Finally, we demonstrate a future for securing life-critical biomedical devices that presents novel clinical capability research concepts for biomedical device cyberattack detection, prevention, and response.

Table of Contents

1 Introduction

This study presents a detailed analysis of the challenges and potential solutions to securing and patching life-critical biomedical systems. It presents research and findings from thorough literature review, clinical and industry surveys, exploratory proof of concept research and development, and biomedical device analysis and teardowns. These findings are synthesized to present recommendations for future research and development to advance the science of securing life critical systems. This section presents our scope of devices, scope of research, and detailed outline of our study.

1.1 Our Scope for Life-Critical Systems (LCS)

In our study, we focused on biomedical systems, which may be composed by one or more devices, that had four criteria:

- Connection capability: Device must be capable of connecting (e.g., wired, wireless) to another medical or non-medical product, or to a network, or to the Internet.
- Potential for harm: Cybersecurity incidents affecting the device could directly result in harm to one or more patients.
- Patchability: The device's software or firmware must be patchable.
- Life-Critical: The device's function for a patient is deemed life-critical by the medical professionals that use or prescribe them.

We describe systems meeting this criteria as life-critical systems (LCS). This scope includes implantable (e.g. pacemakers) and wearable medical devices (e.g. continuous glucose monitors, closed loop insulin pumps) as well as healthcare facility-based equipment and infrastructure (e.g. infusion pumps, CT scanners).

1.2 Patch Verification and Validation

Software verification is the process of checking that software meets the design specification by examining software development artifacts such as documentation, system models, and code. *Software validation* is the process of testing software during or at the end of the development process to determine whether it satisfies customer expectations and business requirements. Currently, patch verification efforts are not as rigorous as initial software development verification efforts.Rigorous patch verification is assumed by many biomedical device manufacturers to be unnecessary as patching is simple software maintenance that does not require specification or rigorous verification. However, depending on the nature of the patch, the risk of possible patient harm, and how it affects or adds functionality, this assumption is invalid.

For stakeholders of LCS, FDA-mandated software verification entails confirmation by examination and provision of objective evidence that the specified requirements have been fulfilled [1]. For software and firmware patches, this includes verifying that any change in the system, whether it be

modifying existing functionality or adding new functionality, achieves the specified requirement(s). We can delineate the *types of patch validation* as:

- Validating the integrity and authenticity of a patch.
- Validating that a patch can successfully and safely deploy to the target system.
- Validating the patch does not break critical functionality, especially any functionality supporting the medically intended use of the device.
- Validating that the patch corrects the targeted defect(s) accurately while preserving functionality, and minimizing computational overhead, in the case of a patch that fixes vulnerability and bugs.

Further details on the state of patch verification approaches these types of patch validation are presented in Section 2.

1.3 Labeling Format

The key findings in our study in Sections 2-4 are presented by a unique labelling format which is defined as following: L findings are literature findings, C is for clinical stakeholder findings, I is for industry stakeholder findings, E is for exploratory research findings, and T is for teardown based findings. For example E2 would be the second finding of the exploratory research and development effort. These findings are synthesized to present recommendations in Section 6.

1.4 Detailed Outline

Below is the detailed outline of our study:

- Section 2: A literature review to represent the extant state of biomedical cybersecurity, with novel contributions.
- Section 3: An IRB-approved clinical survey results to capture key stakeholder perspectives.
- Section 4: Exploratory research and development as well as teardown findings to demonstrate the feasibility of research concepts.
- Section 5: A reference matrix of all research findings.
- Section 6: Our synthesized feasibility study findings.
- Section 7: A model for scoring the complexity of high-assurance patch validation.
- Section 8: A future for novel clinical cybersecurity capabilities.
- Section 9: The conclusion of the study.

2 Background

It was important that our research study focus on capabilities that did not yet exist in the market. To ensure we understood the broader landscape we performed an exhaustive literature review. The literature review process was focused on generating a broad-level characterization of the problem space around securing and patching medical devices. A series of questions guided and scoped the literature review, and the findings from the literature most salient to its answer(s) are stated below as major section headings. Subsections cluster approaches identified in the literature according to how they relate to and address the top-level question.

Novel contributions for this section include: an updated threat model for biomedical devices that synthesize multiple works on attacking biomedical devices to present a master threat vector model for red-teaming (Table 1), a summary of security mechanisms presented by the cited literature to date (Table 2), and several findings on the state of security mechanisms and tools for securing biomedical devices.

2.1 What kinds of life-critical systems exist? What cybersecurity rules govern them?

Implantable medical devices (IMDs) and *wearable medical devices* (WMDs) are rapidly proliferating into consumer and clinical markets [2] in support of preventative [3] and corrective medicine [3, 4, 5] use cases. A broad definition of IMDs proposed by Hansen and Hansen [6] is comprehensive across the literature reviewed in this report. They define IMDs as any implanted device that "treats some underlying medical condition, enhances the function or appearance of some part of the body, or provides a previously unrealized ability" [6]. Prominent examples of medical IMDs include pacemakers [6], cardiac defibrillators [7], prosthetic limb control systems [8], cochlear implants [6], spinal cord stimulators [9], brain-machine interfaces [10], and retinal prosthesis [11]. Taken together with the services that support them, such as mobile apps, in-home base stations, and web services, these kinds of IMDs are examples of *life-critical systems* (LCSs). Other non-LCS IMDs included in Hansen's definition of IMDs [6] such as implanted RFID tags [12] or dynamic LED tattoos [13], are usually elective in nature, and often involve implantation in more superficial locations on the body [6].

Wearable devices are both more numerous in the consumer market and varied in their application [14, 15] compared to implanted devices. WMDs, as a subset of wearable devices, are best placed into two groups *regulatory WMDs* [15], which actuate to modulate and regulate various biological systems; and *monitoring WMDs* [14], which focus on providing feedback, usually in real-time, to patients, their families, and medical staff. Regulatory WMDs are certainly the more life critical of the two due to their abilities to actively actuate, usually on biological systems [15]. Example regulatory WMDs include devices capable of hormone regulation [16, 17, 18], artificial external wearable pancreas devices for blood glucose regulation [18], and seizure prediction and prevention devices [19].

Monitoring WMDs, especially diabetes-related such as glucose point-of-care meters and Continuous Glucose Monitors (CGM) should still be considered as LCS when they monitor for dangerous

situations and alert cognizant parties allowing for timely intervention. Hacked or false readings for diabetes-related monitoring WMDs may lead to life-threatening insulin dosage for type 1 patients. Other examples of monitoring WMDs exist in the market, including myocardial ischemia monitors (heart monitors) [20, 21], epileptic seizure detection systems [19], alertness monitoring (drowsiness detectors) for night crews using watches [22] / EEGs [23] / headbands [24], stroke patient rehabilitation feedback and safety monitoring [25], long-term heart rate variability monitoring for patients with various disease conditions (e.g. bipolar disorder) [26], and sleep apnea / sleep disorder monitoring [27]. As with IMDs, it is important to recognize that the WMD itself is often accompanied by other technology including mobile apps and web services that work in tandem to gather, aggregate, report, and optionally act on medical data. Taken together, these constitute LCSs.

Distilled literature finding(s):

- L1: It is important to recognize that LCS and biomedical systems are often not a single embedded device. Consequently, a comprehensive framework for securing and patching LCS should consider embedded systems, mobile devices, and web services.

2.1.1 FDA Classes of Medical Devices

The FDA categorizes a medical device into Class I, II, or III — based on their risk and the regulatory controls necessary to provide reasonable assurance of safety and effectiveness. Class I poses the lowest risk, and III the highest[28]. Everything from tongue depressors, patient scales, and elastic bandages to smart pacemakers, continuous glucose monitors, and smart neural implants are classified under the FDA system.

47% of existing medical devices fall under Class I category[29], and 95% of these devices are exempt from regulatory processes. 43% of medical devices are Class II. Class III medical devices compose 10% of the market, and these devices typically sustain or support life, are implanted, or offer potential "unreasonable" risk of illness or injury, and include breast implants, pacemakers, continuous glucose monitors, and insulin pumps.

2.1.2 Summary of US Laws, Regulations, and Guidance for LCS Cybersecurity

This section summarizes the extant laws, standards, and regulations related to cybersecurity of life-critical systems at a high level. Such information can commonly be found on the FDA's cybersecurity portal[1].

2.1.2.1 Summary of Laws and Regulations

The FDA's Center for Devices and Radiological Health (CDRH) regulates firms who manufacture, repackage, relabel, and/or import medical devices sold in the United States, and hosts online[2] a

[1] https://www.fda.gov/medical-devices/digital-health/cybersecurity

[2] https://www.fda.gov/medical-devices/device-advice-comprehensive-regulatory-assistance/overview-device-regulation

comprehensive overview of all applicable regulations for medical devices [30], covering the following Title 21 Code of Federal Regulations[3]. CFRs relevant to this work include:

1. The Quality System Regulation (QS regulation) - 21 CFR Part 820
2. The Medical Device Reporting (MDR) - 21 CFR Part 803
3. Premarket Notification 510(k) - 21 CFR Part 807
4. Premarket Approval (PMA) - 21 CFR Part 814
5. The Reports of Corrections and Removals to Medical Devices - 21 CFR Part 806

We summarize the dictated requirements from these CFRs below:

- FDA recommends validating all software changes made to address cybersecurity vulnerabilities per 21 CFR 820.30, [31], and [32].
- FDA 510(k) or PMA process approval is generally not required for cybersecurity patches, unless the change significantly alters the safety or effectiveness of the device per [33], 21 CFR 814.39 and/or 21 CFR 820.3(e & j).
- FDA does not require reporting or coordination of cybersecurity patches to medical devices that have *controlled risks* to patient health per [34].
- FDA does require reporting or coordination of cybersecurity patches to medical devices that have *uncontrolled risks* to patient health per [34].
- MDMs bear the responsibility for ensuring the safety and effectiveness of medical device software per [33].
- FDA requires reporting of any incidents or malfunctions of medical devices that post a significant risk of death or serious injury, which includes cybersecurity vulnerabilities and incidents, per 21 CFR Part 803, which defines the Medical Device Reporting program requirements.
- Regulations applicable to medical devices are supported by FDA recommended guidelines and standards from organizations such as IEEE and ISO [35].

In [36] a summary of international regulations and requirements governing biomedical devices is provided.

[3] 21 CFR Chapter I, Series 800 is specifically the set of regulations for medical devices

Distilled literature finding(s):

- L2: Manufacturers are responsible for making an argument and collecting evidence for demonstrating the safety and effectiveness of their devices. However, self-examination is a poorly regarded assurance mechanism for security. FDA certified 3rd-party device certification labs could provide higher levels of assurance in Premarket security design and post-market patch validation.

2.1.2.2 Summary of Guidance

- The FDA regularly updates the set of standards (FDA Recognized Consensus Standards [37]) it recognizes for use in support of regulatory compliance.
- The FDA's published General Principles of Software Validation (GPoSV) supports the QS regulation's requirements for establish a software validation system of medical devices by defining roles and expectations of software planning, design, assessment, validation, verification, testing, configuration management, and support[32].
- The GPoSV recommends patch validation efforts be conducted "*throughout the entire software lifecycle*" and that "*[w]henever software is changed, a validation analysis should be conducted not just for validation of the individual change, but also to determine the extent and impact of that change on the entire software system*"[32].
- The GPoSV clarifies that when medical devices contain third party software (e.g. libraries, operating systems, and drivers that FDA calls "*off-the-shelf software*"), the medical device manufacturer bears the regulatory responsibility to assess the adequacy of the third party's activities and determine what additional efforts are needed to establish software validation for the device manufacturer's intended use.
- The 2016 Postmarket Management of Cybersecurity in Medical Devices (PMCMD) guidance offers a risk assessment methodology focused on risks of patient harm for evaluating medical device cybersecurity vulnerabilities.

Additionally, the FDA's cybersecurity webpage [38] hosts security alert communications for medical devices, ways to report cybersecurity issues to the FDA, workshops and webinars on cybersecurity policies, and resources from other major players such as MITRE Corporation, the Medical Device Innovation Consortium (MDIC), and the Healthcare and Public Health Sector Coordinating Council (HSCC).

2.1.2.3 Pending Draft Regulatory Requirements

To address existing cybersecurity and testing gaps, support 3rd party testing, and to improve the quality of software risk assessment in medical devices, the FDA has published updated 2018

draft guidance on Content of Premarket Submissions for Management of Cybersecurity in Medical Devices. It recommends that [39]:

- Device design documentation should include detailed system diagrams that clearly explain how:
  - validated software updates and patches will be implemented throughout the device life-cycle.
  - Details system diagrams that illustrate all interacting digital components that includes all details about: network(s), architectures, flow, state diagrams, interfaces, components, assets, communication pathways, protocols, network ports, security controls, authentication mechanisms, users' roles and levels of responsibilities, and cryptographic methods.
- Device risk assessment documentation should include system-level threat models, a list of all known cybersecurity vulnerabilities in the system and it's components, clinical hazard assessments, cybersecurity controls mitigations with documented justifications, and testing reports.
- Device labelling for customers provide details on relevant security information such as, but not limited to:
  - device instructions on implementing cybersecurity and hardening controls,
  - A descriptions of features that protect critical functionality — even when the device's cybersecurity has been compromised,
  - descriptions of backup, update, and restoration features,
  - A Cybersecurity Bill of Materials (CBOM), which includes a list of commercial, open source, and off-the-shelf software and hardware components to enable device users to effectively manage the cybersecurity risks of the system.

The National Telecommunications and Information Administration (NTIA) has supported FDA's efforts and recently concluded in October 2019 an industry pilot effort for Software Bill of Materials (SBOM), and has found that in healthcare they have demonstrated to be viable for medical device manufacturers and healthcare delivery organizations to manage the software of biomedical devices [40].

However, market compliance for CBOM and Premarket threat model requirements will take time. The first post-market guidance on cybersecurity was released in 2014 by the FDA, and compliant devices are just now entering the market in 2019. With the Premarket guidance in draft status, it will likely be several years until Premarket cybersecurity guidance compliant devices are available. Even still, there will be a very long tail of insecure devices deployed in healthcare settings due to several other factors [41].

Clinical stakeholders and patients are sometimes very aware of the baseline injury and death rates caused by medical devices, as "across all types of medical devices, more than 1.7 million injuries and nearly 83,000 deaths were reported to the FDA over the last decade" [42]. Given these

statistics and the technology challenges discussed thusfar, we posit that it is difficult to distinguish between normal or cyberattack caused injuries or deaths to patients. This theory is supported by reports of glitches in devices that may appear like cyberattacks to users [43].

The difficulty of detecting cyberattacks is a profound concern for clinical stakeholders and patients who would be on the front line of a biomedical device cyberattack. To gain a more complete understanding, we consider the state of cybersecurity recalls and present an IRB-approved survey of various LCS stakeholder perspectives.

2.1.2.4 Analysis of Cybersecurity Recalls

To help address vulnerabilities the FDA communicates safety advisories[4] and in cases of significant vulnerabilities posts recalls[5]. Recalls are ranked Class I, II, or III an are in the opposite order of risk to patient harm as the medical device classes, and this may cause confusion in the community. Class I recalls may cause "serious injuries or death" if users continue to use the devices in question, e.g. [44].

Recently on June 27, 2019 several models of older insulin pumps were voluntarily recalled by a manufacturer because of cybersecurity vulnerabilities, which was the first ever cybersecurity recall for connected diabetes devices. Prior to that date, FDA had issued device cybersecurity safety communications only for infusion pumps and implanted cardiac device along with their controllers [45].

As of the time of this writing, according to `https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfRES/res.cfm` there have been:

- Class I recalls:
  - 59 medical device recalls due to "software" reasons.
  - 0 medical device recalls explicitly due to "cybersecurity" reasons.
  - 2 recalls due to software "security" reasons.
- Class II recalls:
  - 10 medical device recalls due to "software" reasons, with most of them being "safety-related" software issues.
  - 2 medical device recalls explicitly due to "cybersecurity" reasons.
  - 8 recalls due to software "security" reasons.
- Class III recalls:
  - 93 medical device recalls due to "software" reasons, with most of them being "safety-related" software issues.
  - 0 medical device recalls explicitly due to "cybersecurity" reasons.

[4] `https://www.fda.gov/medical-devices/medical-device-safety/safety-communications`
[5] `https://www.fda.gov/medical-devices/medical-device-safety/medical-device-recalls`

– 32 recalls due to software "security" reasons.

Recalls urge the community to take steps to address the vulnerability, often through upgrading to the latest version of the device. There are many challenges with getting the community to take steps to address vulnerabilities, such as patching or upgrading. We conducted an IRB-approved survey on these challenges and present our results in Section 3.

2.2 What is the state of life-critical system cybersecurity? What safety and security challenges exist?

With the proliferation of IMDs and WMDs, especially as part of life-critical systems, there has emerged a new threat landscape that challenges patient safety and security. Within this context, we section examines the safety and security challenges that exist for today's devices and explores current practices in device design, implementation, and maintenance that medical device manufacturers are employing to mitigate and counter these threats. The examination of the problem offers a lens towards identifying problems and gaps that are ripe for research and development efforts.

2.2.1 Statistics on Life Critical System Vulnerabilities

According to the Department of Homeland Security's Industrial Control System Computer Emergency Response Team (ICS-CERT) security advisory database for medical devices, 57.29% of exploitable vulnerabilities in medical devices are classified as requiring "low skill to exploit". Additionally, 37.85% of these exploitable vulnerabilities require "high skill to exploit". There are many possibilities that may explain the high percentage of low skill exploits for medic devices, such as ineffective or limited software security testing, threat modeling, security controls, or perhaps significant advances in attacker techniques.

Distilled literature finding(s):

- L3: The fact that most medical device vulnerabilities require low skill to exploit indicates there is a gap in vulnerability testing, threat modeling, or security controls of software and patches in medical devices.

2.2.2 (In)security Landscape

There are many security challenges for life-critical systems. Before characterizing the various types of issues LCSs face, it is important to first be clear about the vocabulary of insecurity. To this degree, this report makes use of the following terms. An *asset* in an information system is anything of value [46], just as in finance. Assets by definition are worth protecting, since their deprivation causes a loss of value to their owners. Assets can be both tangible and intangible. Examples of tangible assets within LCS include IMD and WMD hardware [2], any sub-components on the hardware, patient data stored on the devices [6], and the associated software, apps, and networks

that the devices work with to realize their purpose(s) for the end-user patients [47]. Examples of intangible assets in LCS include the intellectual property embedded in the design of the hardware and software operating in the device [47], brand recognition for the vendors/manufacturers [48], and qualities such as peace of mind and trust that a patient might have towards their LCS [48].

This report uses NIST FIPS 200 standard for information security, definition of *threats* as "circumstances or events with the potential to adversely impact [systems] ...through unauthorized access, destruction, disclosure, modification of information, and/or denial of service [of assets]" [49]. Threats are often conflated with *attacks*, but the two are subtly different. Attacks are actions that adversaries undertake to produce the circumstances or events associated with a threat. Attacks can be either *active*, which means it attempts to alter or affect the operation of an asset [50], or *passive*, which means it attempts to exfiltrate information from the asset without affecting it (e.g. wiretapping) [50]. Threats and attacks may be described in the literature as *threat vectors* or *attack vectors* [51], used interchangeably. This nomenclature implies a series of actions exists that *attackers* can undertake to threaten assets. Threat/attack vectors in the context of LCS are numerous and are enumerated below in section 2.2.3.

Attacks are successful if there are *vulnerabilities* in the target system that can be exploited and there are no *countermeasures* [52] in place on the asset to inhibit or prevent the attack. Successfully executed attacks result in *threat realization* [53], more commonly known as a *security incidents*. For more context on threats, vulnerabilities, and countermeasures in LCS, Section 2.2.3 identifies known threat vectors, Section 2.2.4 investigates and summarizes known vulnerabilities associated with LCSs. Section 2.3.4 addresses current practices in LCS device design, implementation, and maintenance that can prevent attacks from becoming security incidents by applying countermeasures throughout the software and hardware development life-cycle.

2.2.3 Threats to Life Critical Systems

This section presents a comprehensive survey of models for understanding the threat landscape to LCS as well as actual known threats.

2.2.3.1 Models for Understanding the Threat Landscape

The threat landscape for LCS is varied according to the types of assets in the LCS. While many threat modeling techniques and tools exist, such as STRIDE [54] and BSIMM's Attack Modeling framework [55], this report does not focus on techniques for building threat models so much as the outcomes from threat modeling efforts for LCS in the literature. LCS threats are categorized according to how they relate to specific types of assets and device capabilities.

Recent work by Kintzlinger and Nissim [2] surveyed attack vectors related to IMD and WMD LCSs, which they lump together as *personal medical devices* (PMDs). They created data and attack flow diagrams by identifying the key actors, components, and the types of interactions that occur within a PMD LCS. As an example, Fig. 1 shows the attack flow diagram for an insulin pump. The diagram identifies an ecosystem (large dotted box) of internal components and the connections those components have to external devices in other parts of the LCS, such as a mobile app or cloud web service. In this case the insulin pump, a glucose meter, and a glucagon sensor all have the ability

to interact with external devices. The small numbers are different attacks, which are discussed and summarized later in Table 1.

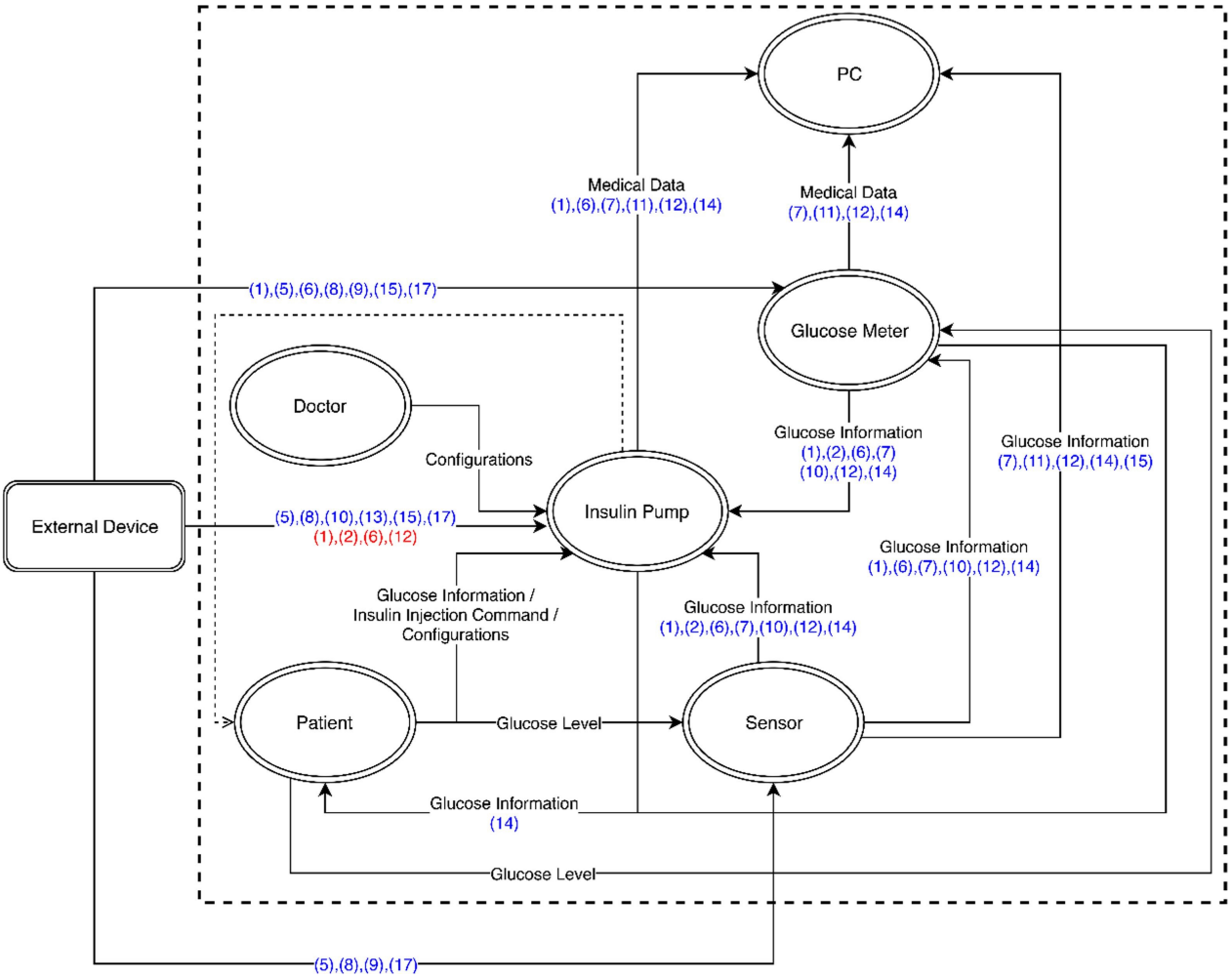

Figure 1: Insulin pump LCS data and attack flow diagram, created by [2]

Attack diagrams of this kind are useful for understanding the security posture of targets in the LCS and for prioritizing countermeasures to be applied in the system architecture. Overall, Kintzlinger and Nissim created attack diagrams and surveyed attack vectors from the literature for the 10 most common PMD LCSs related to 5 disease conditions [2]. Fig. 2 shows the taxonomy of devices they examined. 50% of the attack vectors they investigated relied on the ability to remotely connect to a PMD, while 61% required physical proximity to a PMD [2].

Three other similar surveys [56, 47, 57] established high level threat models and attack vectors for LCSs. Camara, Peris-Lopez, and Tapiador [56] examined threats to IMD LCSs by looking at the types of adverse events that occur in these systems, differentiating them according to threat actors,

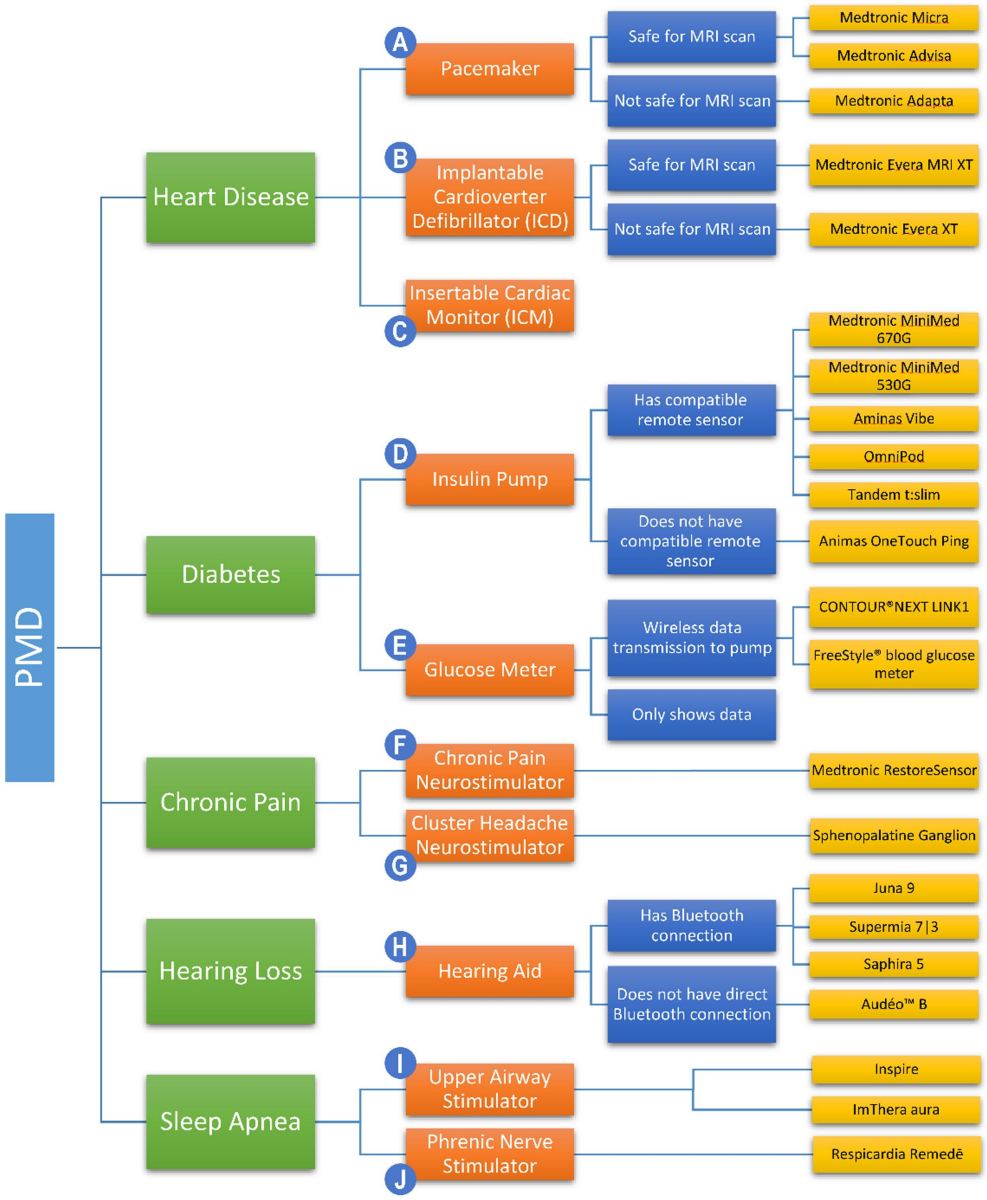

Figure 2: PMD by disease condition from [2]

and then applying the STRIDE [54] threat modeling approach to distill and categorize threats into several high-level classes. Coppolino et al. [47] focused on hardware-assisted security capabilities and in this context enumerated many threat vectors from edge to cloud in LCS architectures. Altawy et al. [57] analyzed both threats and solutions to IMD security. To facilitate compilation in this report, findings from all of these works were synthesized together with those of Kintzlinger and Nissim [2] into Table 1.

ID#	Name	Type	A	B	C	D	E	F	G	H	I	J
1	Resource Depletion	D	X	X	X	X	X		X	X	X	X
2	Dose Change	T				X		X	X			
3	Actuation	T	X	X		X		X	X		X	X
4	Spyware	I	X	X	X	X	X	X	X	X	X	X
5	Electromagnetic Interference	T/D	X	X	X	X	X	X	X	X	X	X
6	Jamming	D	X	X	X	X	X	X	X	X	X	X
7	Man in the Middle	S/T/I	X	X	X	X	X	X	X	X	X	X
8	Firmware Update	T/R/D	X	X	X	X	X	X	X	X	X	X
9	Measurement Disruption	D	X	X	X	X	X	X		X	X	X
10	Alert Attack	D	X	X		X		X				
11	Wireless Protocol	S/T/I/D/E	X	X	X	X	X	X	X	X	X	X
12	Wireless Replay	T/D	X	X		X	X	X		X	X	X
13	Revealing PMD's Presence	I	X	X	X	X	X	X		X	X	X
14	Data Manipulation	T/R/E	X	X	X	X	X	X	X	X	X	X
15	Ransomware	T/D	X	X	X	X	X	X	X		X	X
16	Sound Eavesdropping	I								X		
17	Sonic Gun	D	X	X	X	X	X	X	X	X	X	X
18	Impersonate Programmer	S	X	X	X	X	X	X	X	X	X	X
19	Impersonate Component	S	X	X	X	X	X	X	X	X	X	X
20	Malicious inputs	T	X	X	X	X	X	X	X	X	X	X
21	Disclose medical info.	I	X	X	X	X	X	X	X	X		X
22	Track Patient	I	X		X		X					
23	Determine Device Info	I	X	X	X	X	X	X	X	X	X	X
24	Drain device battery	D	X	X	X	X	X	X	X	X	X	X
25	Reprogram/Reconfigure	T/E	X	X	X	X	X	X	X	X	X	X
26	Switch off Device	D/E	X	X	X	X	X	X	X	X	X	X
27	Randomness Attacks	S/T/R/I/E	X	X	X	X	X	X	X	X	X	X
28	Direct Memory Access	T/R/I	X	X	X	X	X	X	X	X	X	X
29	Default Authentication	S/T/E	X	X	X	X	X	X	X	X	X	X
30	Reflection	S/I/D/E	X			X	X			X		
31	Code-reuse attacks	T/E	X	X	X	X	X	X	X	X	X	X

Table 1: LCS Threat Vectors synthesized from [2], [47], [57], and [56]. Attack (2) from [2] was changed from *Insulin Dose Change* to *Dose Change*. Attack (3) was changed from *Defibrillation* to *Actuation*. *Denial-of-Service* (Attack 6) was changed to *Jamming* to better distinguish this attack type from other DOS attack vectors. Wireless USB Dongle was changed to *Wireless protocol* (Attack 11) to better clarify that these attack vectors are at the communication level. All changes were made to generalize attack vectors to all types of LCSs and to allow for integration with the work from [56].

To fuse the works together into one table, a series of transformations were applied to the underlying works. First, the original 17 attack vectors from [2] were examined and their details and description were generalized, since some were overly specific to a certain type of device. The intention of the generalization effort was to ensure that each attack could be applied to any type of device. For example, Attack 2 in Table 1 was originally listed as *Insulin Dose Change* in [2] and only applied to PMD D (Insulin Pumps). In our table, the word *Insulin* was dropped and the attack vector was reevaluated against the other PMDs in the taxonomy. This covered a weakness of the prior work. Other changes of this kind are listed in the description of Table 1. The second transformation applied to fuse the works together required us to examine the attack vectors from [56], [47], and [57] in the context of the taxonomy to determine if attacks 1-17 covered them or if additional attack descriptions were necessary for completeness. Based on a review of the attack vector descriptions from across the works, attacks 18 to 26 from [56], attacks 27 and 28 from [47], and attack 29 from [57] were not covered in attacks 1-17. Hence those attacks were added to Table 1. In the analysis of these additional works, multiple repeats of other prior attacks were observed - adding support for the preeminence of this iteration. Following the methodology in [2], the newly listed attacks were examined to determine if they applied to the PMDs in the taxonomy.

Most of the threat vectors were generally applicable to all PMDs, but a few such as Attack 21 and 22 only applied to a few device types. IoT devices are often fertile targets for reflective attacks that seek to bounce challenge messages (e.g. ICMP, SSDP, SNMP and SNMP public community string)[58] back to the originator or other targets to create a flood of unwanted traffic. Considering that this type of attack is very relevant for mobile apps and base stations that interface with PMD's, we chose to add this attack to Table 1. Attack 31, a code-reuse attack comes from [47] which includes attacks such as return or jump oriented programming, control flow hijack, or buffer overflow attacks that chain together existing code fragments to implement malicious logic. Finally to finish the table, the recommended STRIDE threat assessment approach, from [56], was applied to each of the attack types not already categorized. STRIDE is a keyword that helps enumerate **S**poofing, **T**ampering, **R**epudiation, **I**nformation disclosure, **D**enial of service and **E**levation of privilege threats.

In [59], Alexander, Haseeb, and Baranchuk closely examine security recall information from [60] to detail a case study of security issues in an Abbott pacemaker. The case they describe relates to Attacks 6 and 24 from Table 1. By sending a large amount of radio traffic in the frequency band the pacemaker operated in to communicate with its associated home system app an attacker could make the pacemaker become unresponsive to radio telemetry from an interrogation device [59]. Another attack allowed actors within a few feet of the device to drain its battery using similar jamming techniques. In response, the company developed a firmware upgrade patch that addressed the problems. The firmware upgrade took approximately 3 minutes to complete and had its own risks - such as loss of device function, change of pacing behavior, and failure to update. While no misuse was documented, the potential risk analysis of the harmful potential effects speaks to the highly impactful nature of Attack 8.

Distilled literature finding(s):

- L4: It is clear that LCS present a large attack surface with many possible threat vectors. Unlike industrial systems with many assets and actuation points, the *impact surface* of an LCS that can cause patient harm is often limited and focused. These focused biological processes, should be monitored for anomalies (e.g. IDS/IPS) during normal operation and while patching.
- L5: For an LCS, internal-devices are low-powered and access requires physical proximity, but internal-LCS and external components are exposed to network and cloud-based infrastructures. Thus when securing and patching internal-devices, these devices should not have any inherent trust in other LCS components.
- L6: Malicious actors sometimes chain together actions that were not foreseen by the original designers. Therefore, in the verification of medical devices security mechanisms, there should be attempts to bypass defenses by a team of technically-competent individuals (e.g. 3rd-party FDA certified assessment labs) that are not part of the original design or development teams.

2.2.3.2 Active Threats to LCS

There are many active hacking campaigns targeting healthcare environments and medical devices due to their importance in critical infrastructure and society. In a recent survey of healthcare organizations across the US, UK, Japan, Germany and China, a staggering 82% of respondents reported that cyberattacks targeted and/or impacted IoT systems (which include biomedical and operational technology systems) in the last 12 months. Additionally the average financial damages and/or incident response costs are reported to exceed $340,000 per IoT cyberattack [61, 62]. Other reports estimate that cyberattacks targeting healthcare IoT systems have increased 300% from 2018 to 2019 [63, 64].

The value and utility of personally identifiable information (PII), protected health information (PHI), intellectual property data, and other protected information are key motivators for criminal attackers. Medical records are often sold in bulk on the darkweb or private forums, where they can be worth up to $1,000 per record. Much of this information can be monetized or further weaponized for higher impact [65]. Databases of patient data have been found for sale in underground forums and the data can be further leveraged for identity theft or custom phishing campaigns [66]. Attackers will also utilize their access, and associated trust relationships, to exploit additional devices or networks.

Criminal actors tend to value exploiting opportunistic targets with poorly secured networks and focus on patient data while espionage campaigns appear driven to acquire medical research or data sets [66]. Attacks perpetrated by criminal actors appear with higher frequency over espionage campaigns sponsored by nation state threats. Additionally, criminal actions have indirectly impacted biomedical devices in recent history [67].

Cyber espionage campaigns, by contrast, seem to prioritize stealing sensitive data and intellectual property associated with cancer research from servers, desktops, and biomedical devices.

Most known advanced persistent threats (APT) such as APT18 [68] and APT41 [69] have been associated with targeting biotech and pharmaceutical organizations using techniques such as advanced spearphishing campaigns and keylogging malware by attacking servers, desktops, and other IT targets. Instead of patient data, they are reported to gather clinical trial data, academic data, and research funding documents. These actors also leverage industries in other verticals, such as manufacturing, that provide services or products to healthcare organizations.

Other such actors are targeting biomedical devices. Most notable, an espionage campaign dubbed OrangeWorm, has been active since 2015 and has primarily leveraged supply-chain attacks to deliver trojanized biomedical devices and equipment to multiple sectors, with a primary focus on healthcare. The campaign's "malware was found on machines which had software installed for the use and control of high-tech imaging devices such as X-Ray and MRI machines"[70]. This campaign's utilizes a custom built backdoor, dubbed `Trojan.Kwampirs`, and allows the attackers espionage-based capabilities on affected systems, but could be trivially modified to cause other impacts such as ransomware. The group, although not considered to be motivated by a nation state, is thought to specifically target victims of interest before establishing further persistence in the network.

Possible directions for future attack campaigns on medical devices and their support infrastructure are concerning. Mirsky et al [71] presents an attacker model focused on deceiving picture archiving and communication systems (PACS) like CT and MRI machines to adversely change patient treatment. By leveraging weaknesses in infrastructure [72] [73], attackers can inject false data into the 3D scan and impact patient treatment regimens. Machine learning in the form of an adversarial network is used to generate the "poisoned data". Motivations for deceiving healthcare professionals range from committing insurance fraud (financial) to harming or holding ransom individual lives.

2.2.4 Vulnerabilities in Life Critical Systems

Known LCS security and vulnerability analysis works focused on IMDs and/or WMDs are examined to augment the LCS threat discussion and attack vector synthesis process. The scope of inquiry included all elements of LCSs from hardware to cloud components. This section describes known vulnerabilities that have been documented in the literature.

Hansen and Hansen created a taxonomy of known vulnerabilities for IMDs [6]. Although specified for IMDs, their work is also applicable to WMDs. The key findings for vulnerabilities are summarized below:

- Their classification scheme takes an asset-first perspective to categorize vulnerabilities. In their taxonomy, assets can either be patient- or component-oriented. Within the component oriented groups they differentiate between components in two ways:

  1. where the target component resides, and
  2. the kinds of capabilities the component has, i.e. *processing*, *actuation*, *communication*, or *sensing*.

- On the component side, the Hansen model offers a topology to describe the areas components can be affected, as "IMD - target IMD", "IMD - other IMD", and "external device". To translate: target IMD/LCS is the LCS device itself, other IMD/LCS is a component of the LCS connected to but not part of the LCS itself, and external devices are those that are not part of the LCS.
- On the patient side, the Hansen model proposes key biological systems as target assets including *circulatory*, *gastrointestinal*, *genitourinary*, *immune*, *musculoskeletal*, *reproductive*, *sensory*, and *skin/cosmetic*.
- The Hansen model offers causal and effectual elements to categorize vulnerabilities:
  - Causal elements include the *threat vector* that seeks to exploit a vulnerability, the *activity* associated with the vulnerability, and the *patient state* that a patient must be in for the vulnerability to be exploited.
  - Effectual elements include the asset qualifiers above and the *permanence* of the negative outcomes if the vulnerability is exploited.

Williams and Woodward [3] note a number of reasons why PMDs are particularly prone to vulnerabilities. The key findings of their work are summarized below:

- Often LCSs use legacy operating systems and software.
- IMDs, are rarely patched by vendors or when they are patched it takes significant longer than other fields [74].
- Security features are often added after critical features on PMDs, leading to incongruous design interoperability or poor protective capabilities.
- LCSs often incorporate web services as middleware to existing clinical systems, particularly EMR systems. These web services often do not follow good security practices and rely on security through obscurity.
- Compromised PMDs can easily act as gateways for other attack vectors in LCSs, potentially threatening other medical systems or other PMDs.
- Trade-offs exist between security and ease of interaction for emergency response. Usually emergency response is prioritized over security.
- PMDs have limited power and computing resources similar to low-powered IoT devices, which put hard bounds on protection mechanisms.

Other works by [57] add a few other challenges that PMDs face, including:

- Adoption of PMD security mechanisms can be held back by bureaucratic problems, particularly for IMDs since the devices often must undergo re-certification testing by regulatory bodies.

- During lengthy review, security mechanisms may be invalidated due to a changing threat landscape or other new vulnerabilities.
- Biological operational environments can be limiting, preventing IMDs from adopting certain security features if they emit too much RF radiation, do not fall within certain power/heat dissipation constraints, or take too long to respond due to complex processing. Over-dissipation can result in "implant induced coagulation and/or allergic foreign body response" [75].

Additionally, Rios and Butts in [76, 77, 78] "analysed seven pacemaker products from four different vendors and discovered that they use more than 300 third-party libraries, 174 of which are known to have over 8,600 vulnerabilities that hackers could exploit in pacemaker programmers". We summarize their findings as:

- The assessment and patching of third party code in LCS has been an ubiquitous significant gap, despite the guidance in the FDA GPoSV instructing MDMs to test and be responsible for such code.
- There is a significant secondary related gap that healthcare delivery organizations are completely unaware of the software and third party software in the medical devices in their environments, and thus are unaware of what can and should be patched.

Finally, Rushanan et al. [79] add a few more challenges not covered by the others reviewed to this point, namely:

- Reproducibility for security threat evaluation, vulnerability analysis, and mitigative measure effectiveness is limited by a lack of access to medical devices among security researchers, particularly IMDs.
- Access limitations are primarily based on acquisition difficulty and operational environment simulation integrity issues[79].
- Constantly changing vendor product lines, patches, and hardware manufacturing changes make vulnerability testing across product lines (which is necessary due to code-reuse) prohibitively challenging — especially for discontinued devices that are still in use in or by patients.
- Measurement noise can be a problem when using physiological outcome values as inputs to security and privacy mechanisms [79]. This can lead to a reduction in mechanism accuracy - for instance using ECG values as a random number generator, a tactic often employed by IMDs [80], may not provide sufficient entropy for the cryptographic mechanisms which rely on this information [79].

Leveraging vulnerabilities caused by these gaps, the community surprisingly has found productive uses for exploiting security flaws in medical devices! Security flaws have enabled a practice of Do-It-Yourself (DIY) hacking of medical devices [81]. The goal of DIY medical device hacking is to

get faster access to technological solutions that improve patient quality of life, while mass-market offerings are still pending clinical trials or regulatory checks. However, most people engaging in such hacking do not have the knowledge or skills to verify the patches being applied to their devices. They rely on an open community of volunteer developers and hackers to add features and fix bugs. While the patch is free, the cost of devices that support such hacking has skyrocketed on second-hand marketplaces. Many of these devices are in legacy status or completely discontinued.

Finally, in our review of the literature, there are two primary points of good news. First, many vulnerabilities of PMDs are prone to rely on physical proximity [57, 6, 82] which has been a long standing inhibitor of attack vectors [79]. Second, the interest and research works focused on PMDs are accelerating more as these devices proliferate.

Distilled literature finding(s):

- L7: Transparency of hardware, firmware and software components is an important need for patch and configuration management, and this gap inhibits healthcare organizations' efforts to manage cybersecurity risks to biomedical systems. This may be addressed by the cybersecurity-bill-of-materials proposed by the draft Premarket guidance [39].
- L8: Systems that make up LCS and biomedical systems often use vulnerable legacy operating systems, software, and components which should be considered in comprehensive security and patching strategies. Compromised legacy systems can act as easy gateways for other attack vectors to threaten other biomedical and LCS targets.
- L9: Medical device manufacturer assessment and patching of 3rd party components in LCS is a ubiquitous, significant gap despite longstanding FDA requirements to do so. Many vendors have historically offered LCS with known unpatched vulnerabilities in the 3rd party components.
- L10: Reproducability of bug and vulnerability reports as well as threat evaluation findings is a challenge for LCS [79] due to a lack of access to devices for 3rd party security researchers and the velocity of LCS change in the market.

2.2.5 Medical Device Risk Scoring and Management

Manufacturers of medical devices have to evaluate the risk of post-market updates and patches. The 2016 Postmarket Management of Cybersecurity in Medical Devices offers risk assessment recommendations on how to determine if the patch poses risks to patient health which may require re-certification, reporting and/or coordination with the FDA: "For a small subset of actions taken by manufacturers to correct device cybersecurity vulnerabilities and exploits that may pose a risk to health, the FDA would require medical device manufacturers to notify the Agency"[34]. The goal of such risk analysis is to determine if the risk of patient harm is controlled or uncontrolled. If risks are deemed to be uncontrolled, then manufacturers must inform FDA per MDR regulations with certain exceptions. The assessment of risk in the guidance is based on an evaluation of the likelihood of exploit, the impact of exploitation on the device's safety and essential performance, and the severity of patient harm if exploited (see Figure 3) [83]. Impacts to patient privacy are not included in the patient harm risk assessment methodology, but are covered by HIPAA.

Stine et al. [84] have developed a cyber risk scoring system for use in hospital acquisition processes. Ease-of-use, low cost, and easy-to-understand results are priorities for use of this risk ranking scheme by hospital staff and end users. They use a questionnaire-driven approach to combine a physician's worst-case assessment of the potential of a medical device to impact a patient with an assessment of the device's security features. Similar rating systems exist to evaluate the security/privacy posture for consumer IoT devices using automated scripts [58]. Risk management processes are essential for healthcare providers and insurance companies to make effective choices when comparing different medical device offerings in support of their operations. Within the last

decade, healthcare organizations have adopted the Manufacturer Disclosure Statement for Medical Device Security (MDS2) in their procurement requirements [85]. The MDS2 form provides medical device manufacturers with a means for disclosing to healthcare providers the security related features of the medical devices they manufacture. The MDS2 also requests details that are noteworthy to patch validation such as the manufacturer's roadmap for 3rd party component support in the device lifecycle. Williams et al. [3] recommend a systemic perspective that includes technical controls, governance, resilience measures, consolidated reporting, context expertise, regulation, and standards to deal with a multi-faceted problem of medical device security.

Distilled literature finding(s):

- L11: To facilitate risk management efforts, new scoring methods are needed to evaluate the complexity of assuring correct operation of an LCS patch that meets all user's needs and intended uses. Such a scoring system should be able to accommodate the entire embedded system stack including firmware and hardware and all input/output interfaces, such as network, sensors, and actuators for effective estimation of patch validation complexity. Such a scoring system should also accommodate the assurance needs of LCS end-uses and care-providers for patches

Figure 3: "Evaluation of Risk of Patient Harm" model from 2016 Postmarket Management of Cybersecurity in Medical Devices [34].

2.2.6 Engineering Guidance and Best Practices

2.2.6.1 Scope and Applicability

FDA's guidance documents provide manufacturers of medical devices with non-binding recommendations. This means that the best practices are to be considered as suggestions or recommendations, but not required. Often an industry-wide status quo determines appropriate rigor in Premarket certification. Current cybersecurity related Premarket and post-market guidance from FDA is in draft stage and does not require implementation. Furthermore, FDA always seeks to take a *least burdensome approach* to resolve issues raised in a guidance document. FDA guidance on software applies to any software component, part or accessory in a medical device [1]. This includes firmware that controls a medical device, dedicated hardware/software medical devices, or accessories to medical devices composed of software. The guidance applies regardless of software abstraction level.

Bhunia et al. [86] identify six levels of software on top of digital logic hardware. These include Level 1: Micro architecture; Level 2: Instruction Set Architecture; Level 3: Operating System Machine; Level 4: Assembly Language; Level 5: Problem-oriented Language; and Level 6: Object-oriented Programming language. Manufacturers focus on improving the quality of software in Levels 5 and 6, but do not have the resources or skills to analyze software in their supply chain at Levels 1-4. Per FDA Premarket guidance [39], manufacturers only document cybersecurity design features and risk management strategy to demonstrate the reduction of risk to an acceptable level.

Per statements from FDA [87] a new Center of Excellence for Digital Health [38] is being planned. This center will have greater emphasis on assessing cybersecurity of medical devices and would evaluate and recognize 3rd-party certifiers, and support a cybersecurity unit to complement the advances in software-based digital health devices [87] FDA's digital health action plan [88] outlines specific recommendations for software as a medical device (SaMD), mobile medical applications, and wireless medical devices. Most notably the digital health software pre-certification (Pre-Cert) program is geared towards regulatory oversight of software development processes and organizational excellence as opposed to a product-focused view. Cybersecurity responsibility is one of the five excellence principles in this proposed framework. Note that while FDA maintains oversight, post-market validation of all changes to software design, code, and their testing is the responsibility of the device manufacturer. The manufacturer also assumes responsibility for any software procured from its supply chain. As a result, FDA certified 3rd-party reviewers working closely with manufacturers would allow cybersecurity analysis of next generation medical devices for digital health initiatives to be reliable, detailed, and scalable.

2.2.6.2 Federal Best Practices and Recommendations

Draft Premarket cybersecurity guidance from the FDA [39] recommends the application of any standard with the word *security* in its title in the FDA recognized consensus standards database [37]. As of this writing, the included standards in the database span the following broad cybersecurity topic areas:

- Risk management for IT networks with medical devices [83].

- Industrial communication network and system security.
- Software cybersecurity for network-connectable products.
- In-vitro diagnostic instruments and software systems.
- Principles of medical device cybersecurity.
- Vulnerability handling and disclosure.

Specifically for software development, the recognized standards span the following additional topic areas related to cybersecurity of medical devices:

- Medical device software life cycle processes [89].
- Validation of software for regulated processes.
- Classification of defects in health software [90].
- Use of AGILE for medical device software development.
- Systems and software assurance, including assurance case development.
- Software in programmable components.
- Software in lab instruments and data management systems.

FDA has shown a long standing interest in the development of high-confidence medical device cyber physical systems [91, 92, 93, 35]. However, current FDA recommended consensus guidance documents and standards do not reflect the rigors of verification, validation, and certification necessary to procure high confidence medical devices. Many of the research challenges identified a decade ago in the NITRD research report [92] still remain relevant. One of those challenges is to be able to "*provide sufficient high-quality evidence that any system design or component change is appropriately accounted for in all system components, such as assumptions, information, timing, risks, etc*". The report also goes on to recommend *evidence-based certification technology* for higher confidence when patching or updating medical devices. It states that "*instead of seeking to provide evidence that a process has been planned for and followed, the evidence-based approach seeks to generate independently verifiable evidence that a system satisfies its requirements*"[92] . For example, certification could include assuring safe input/output operations over all operating modes for new driver patch validation. Innovative tools and methods that operate at various levels in the embedded system stack are urgently needed to establish sufficient evidence that various system components and properties are not affected by a design or component change.

2.2.6.3 Ancillary Best Practices and Recommendations

Concerned with the state of cybersecurity practices in the healthcare sector, several organizations including the Diabetes Technology Society and the Health Sector Coordinating Council (HSCC) have also released guidance and policy recommendations to address critical gaps in this space.

The Diabetes Technology Society Cybersecurity Standard for Connected Diabetes Devices (DTSec) is a common criteria protection profile [94] with extended packages (basic [95] and moderate [96]) developed through a public, private, and government partnership. The DTSec protection profile includes the *fundamental security and evaluation requirements for connected diabetes devices, including blood glucose monitors, continuous glucose monitors, insulin pumps, and handheld controllers* [94]. The primary threats identified in this protection profile include network access, physical access, malicious firmware or applications, a malicious peer device, and weak cryptography. The protection profile maps specific functional security requirements derived from security objectives that address the threats, demonstrating security engineering activities. Additional security assurance requirements are integrated into the evaluation process through the basic [95] and moderate [96] extended packages. DTSec has designated a set of approved testing labs that evaluate security targets developed by vendors interested in the certification. These labs have certified a total of three products from Ascensia and Insulet manufacturers to date. Cardiac implantable electronic devices do not yet have a similar standard, but recent major publications indicate collaboration between the FDA, manufacturers and professional societies such as the American College of Cardiology and Hearth Rhythm Society may build one [97]. We posit that risk discovery efforts need to be based on criteria tailored to medical device embedded systems that can be objectively tested by 3rd-party certifiers and extensively tool supported for increased assurance in Premarket cybersecurity analysis and post-market patch validation. We offer contributions to this challenge in Sections 6 and 7.

The Health Sector Coordinating Council (HSCC) Joint Cybersecurity Working Group (JCWG) is a collective consisting of 200 member organizations that have formed working committees focused on addressing the findings of the Congressionally-created, HHS-appointed 2017 Health Care Industry Cyber Security Task Force [98]. JCWG is a subgroup of the larger Healthcare and Public Health Sector Coordinating Council (HPH SCC) which is a coalition of private-sector and critical infrastructure entities organized under Presidential Policy Directive 21 and the National Infrastructure Protection Plan. HSCC maintains a public-private partnership and collaborates with public health sector government officials. The findings of HSCC's report include six imperatives, multiple recommendations, and action items for addressing the imperatives. More specifically, the report outlines the need to: define and streamline leadership, governance, and expectations across the healthcare industry, increase the security and resilience of medical devices and healthcare IT, develop a technical and security-aware workforce, increase readiness through cybersecurity education, identify means to protect intellectual property, and improve information sharing across industry stakeholders. Imperative 2, increasing the security and resiliency of medical devices and health IT, further directs the securing of legacy systems, improving manufacturing transparency, increasing adoption of the secure development lifecycle, requiring strong authentication practices, taking approaches to reduce the medical device attack surface, and establishing a MedCERT team to respond to device-specific cybersecurity incidents and vulnerability disclosures.

In 2018 JCWG published "Health Industry Cybersecurity Practices (HICP): Managing Threats and Protecting Patients" in partnership with the US Department of Health and Human Services [99]. This document was forged through the collaboration of the U.S. Department of Health and Human Services (HHS), the Health Care and Public Health Sector Government Coordinating Council, the Department of Homeland Security (DHS), and the National Institute of Standards and Technology (NIST). The four volume publication consists of 10 recommended cybersecurity practices, their application to small healthcare organizations, their application to medium and large healthcare organizations, and resources and templates that organizations can use to develop their own policies, procedures, and security posture. The 10 recommended practices include endpoint and service protection, access management, data protection and loss prevention, asset management, vulnerability management, network management, incident response, medical device security, and cybersecurity policies. These practices are based on the threat model presented in the first volume that identifies phishing, ransomware, loss or theft of equipment or data, insider data loss, and attacks against medical devices themselves. Their Joint Security Plan (JSP) published early in 2019 is a more mature document that applies the NIST cybersecurity framework to the healthcare sector [100].

In October of 2019, eHealth Initiative and Foundation (eHI) in collaboration with Booz Allen Hamilton, released "Securing Connected Medical Devices" which discussed recent vulnerability advisories (URGENT/11) released by DHS that did not require physical proximity to compromise a connected medical device. They recommended securing the ecosystem, working across stakeholders to address threats and risks, referenced the JSP as a significant industry resource, being proactive in addressing future risks, and engaging with the HSCC and Health Information Sharing and Analysis Center (H-ISAC) [101].

2.3 What barriers impede securing life-critical systems? How can they be surmounted?

This section distils the pervasive challenges to securing life-critical systems into discrete barriers to focus the research discussion. Below we discuss three significant, discrete barriers to securing life-critical systems and the broader market of biomedical devices:

1. Barriers to software, firmware, and patch testing.
2. Barriers to biomedical cybersecurity attack detection, prevention, and response.
3. Barriers to communicating vulnerabilities and understanding cybersecurity risk.

Afterwards is a focused discussion on capabilities and limitations of existing approaches that address these challenges.

2.3.1 Barriers to Software, Firmware, and Patch Testing

There is a significant gap in testing for safety versus cybersecurity of biomedical devices. This is driven by the different levels of abstraction at which the security and safety implications of a medical device are assessed, which consequentially result in vastly different assurance artifacts. Elmiligi

et al. [102] have identified multiple dimensions of embedded systems security which are missing in CMVP/FIPS 140-2 and Common Criteria (CC) assessments of general IT systems. They identify four different levels of security mechanisms necessary to address 1) programmability levels offered by hardware, firmware, and software; 2) integration level in intellectual property cores, chip or board; and 3) life-cycle phases when the attacks occur. Furthermore, different software/firmware analysis levels may also be more effective and accurate at finding possible attacks. Tabrizi et al. [103] found that for a smart meter IoT, device code-level analysis using symbolic execution was more effective and accurate in finding software interference attacks, compared to design-level analysis using model checking. This result is quite surprising as one would expect the state-space to be much smaller at the design-level and hence produce fewer false positives. By extension, analysis further down the embedded system stack (including firmware and hardware) and all input/output interfaces (network, sensors, actuators) can be more effective and accurate in discovering new attacks. However, regulation and accompanying standards and guidance would likely never mandate such analysis to address this gap without general availability of such capabilities, skills and tool sets. Indeed, manufacturers and regulators need practical approaches to gain security and safety assurances. Given complex software-supply chains and intellectual property rights manufacturers and 3rd-party certifiers alike may never have full access to software source code. A combination of white, grey, and black box testing is required for different software levels in an embedded system. In the post-market phase of a device, there is limited knowledge about the original device requirements and specifications [104]. As a result, validation is often limited to externally observable behaviors of the device. Similarly, without original design specifications, verification of the effects of a patch on unit, integration, and system level contracts is less rigorous.

There is a significant role that technology research and development can play in developing practical advancements for testing biomedical software, firmware, and patches. However, to temper the discussion, due to limits in all vulnerability analysis techniques the only guaranteed way to unravel the full behavior of a software patch is to run analysis in perpetuity. Static, dynamic, and hybrid code analysis techniques suffer from theoretical limitations (e.g. the halting problem and infinite test space) for achieving complete coverage of code behavior. Smart fuzzing techniques [105, 106] have found better success in practical applications compared to computational software verification techniques for program flow analysis like concolic and symbolic execution. However, diverse micro architectures, instruction sets, and customized input/output (I/O) channels hinder high-fidelity virtualized simulations for embedded system operations and hence limit the effectiveness of fuzzing [107]. These aspects of the problem space related to medical devices make generalizable software and patch validation and verification a hard problem. In the absence of computational guarantees, there have been many attempts in prior literature to tackle various aspects of the problem space in specific case studies with medical devices and more generally for embedded systems. A review of literature and published capabilities in the following subsections (2.3.5 and 2.3.6) facilitates a gap analysis. Finally, we offer contributions to addressing this barrier in Sections 6.3 and 7.

Distilled literature finding(s):

- L12: Given complex software supply chains and intellectual property rights, manufacturers and 3rd-party certifiers alike may never have full access to software source code. A combination of white, grey, and black box testing is required for different software levels in an embedded system.
- L13: Dynamic testing of embedded device firmware binaries will require a high fidelity model of all special-purpose hardware components that it uses in the real system. Emulators that support hardware in the loop operation are needed to accommodate diverse and often custom hardware components used in LCS. Such test-beds are essential to further the science of securing and patching LCS.
- L14: Diverse micro architectures, instruction sets, and customized input/output (I/O) channels hinder high-fidelity virtualized simulations for embedded system operations [107].
- L15: There are few available firmware analysis tools that can be applied to LCS architectures (discussed further in Appendix A).

2.3.2 Barriers to Biomedical Cyberattack Detection, Prevention, and Response

There are significant gaps in capabilities for detecting, preventing, and responding to cyberattacks against biomedical devices. This is based on the presented statistics, literature, and findings showing alarming trends for LCS vulnerabilities in Section 2.2.1, threats in Section 2.2.3, and actual malware campaigns in Section 2.2.3.2. A deeper review of literature and published security control capabilities in subsection 2.3.4. Finally, we offer contributions to address this barrier in Sections 6.1 and 6.2.

2.3.3 Barriers to Communicating Vulnerabilities and Understanding Cybersecurity Risk

There is a significant gap in accurately describing and communicating vulnerabilities. While the 2002 General Principles of Software Validation recommend a focus on defect prevention, it is not until recently that comprehensive defect taxonomies for medical device software have become available [108]. The development of the health software defect taxonomy SW19[90], portions of which are also mapped to the common software weaknesses: CWE-884 [109] is a step in the right direction for assessing medical device software. However, discovery of software defects using taxonomies [110] and checklists presents a significant development overhead for software-intensive medical devices. It requires an investment in re-training developers and testers as well as supporting them using software assurance tool sets. The quality of threat models (recommended by FDA [34]), design and code review rigor, and the effectiveness of software assurance tools also influence the efficacy of such testing efforts.

Additionally, gaps in device transparency and threat modelling contribute significantly to the challenges healthcare organizations have in mitigating and detecting cyberattacks against biomedical systems. However, these gaps may eventually be addressed by the FDA's draft Premarket cybersecurity guidance, which calls for CBOM and threat modelling [39].

2.3.4 Existing Security Approaches

This section reviews the current state of the art across IMD, WMD, LCS, IoT, and embedded hardware security. In the initial breadth search several survey papers were identified. These included those by Kintzlinger et al. [2] (also used in Section 2.2.4), Camara et al. [56], Rushanan et al. [79], [57], and Al-Omary et al. [111]. From there, review proceeded by spidering out the many security approaches they identified in their works. Table 2 and the following list of analyzed security mechanisms establishes a security landscape perspective, and characterizes known techniques in a way that facilitates gap analysis towards LCS/PMD security improvement

Using Kintzlinger's work [2] as a base due to its excellent job of categorizing approaches and its recency (2019), we constructed Table 2. The table gives each mechanism a number (#) for easy reference, names the *security mechanism*, and provides a *src* field to identify the mechanism description source. For each approach, examination started with the dimensions of:

- Confidentiality, integrity, and availability (C, I, and A).
- Anomaly detection (AD), access control (AC), device authentication (DA), and encryption (E) security mechanism categories, as found in the original table from [2].
- Audit (AU) mechanisms from [56].
- Finally we added a hardware-based (H) identifier to indicate whether the mechanism was hardware based.

Using this structure, we examined mechanisms 1-21 in the [2] work and followed them through to their origin papers. For these, each approach was assessed to determine if it included audit (AU) perspectives and was based in hardware (H). In some cases, the security mechanism names were slightly adjusted to better label the approach in a few words. Mechanisms 22 and onward were pulled from the original sources as listed in the table. The methodology from [2] was used to categorize each approach according to its salient features. A brief summary of each security mechanism is provided below:

1. **Access Control Lists** refers to work by Spring et al. in [112] that lays the groundwork for a protocol-based implementation of ACLs for PMDs. It attempts to impose access limitations on external devices to the PMD.
2. **Zero-Power Defenses** refers to a suite of mechanisms that seek to provide IMDs with additional security capabilities without using the battery power of the IMD. These include capabilities to sound an audible alarm, the ability to securely exchange keys with an external device,

#	Security Mechanism	src	C	I	A	AD	AC	DA	E	AU	H
1	Access Control Lists	[112]	X	X			X				
2	Zero-Power Defenses	[113, 114]		X	X		X	X	X		X
3	External Cloaker	[115]		X	X		X	X	X		X
4	Proximity-Based Access Control	[116, 56]	X	X			X				X
5	UV Ink Tattoo as a Key	[117, 118]	X	X				X			X
6	Integrated Comm. w/ Cell Phone	[119]			X	X	X				X
7	Subcutaneous Button	[6]	X	X	X		X				X
8	IMD Guard	[120]	X	X	X		X	X	X		X
9	Rolling Code	[121]	X	X				X			
10	IMD Shield	[122, 79]	X	X	X		X	X			X
11	Lightweight Security Protocol	[123]	X	X				X	X		X
12	One-Time Pad Encryption System	[124]	X	X				X	X		X
13	Personal Security Device	[125]	X	X	X		X				X
14	Biometric Identifiers	[2, 56]	X	X			X				X
15	Ghost Talk: Analog Defenses	[126]		X	X	X					X
16	Split-core System Architecture	[127]	X						X		
17	Anomaly Det. Using Bowel Sounds	[128]		X		X					X
18	MedMon	[129]	X	X	X	X					X
19	Proxy Device	[130]	X	X	X	X	X				X
20	Infusion Pattern-Based AC	[131]		X		X	X				X
21	Physical Layer Authentication	[2]	X	X	X			X			X
22	Emergency Access Overrides	[56]			X		X				X
23	Wristband passwords	[56, 79]									
24	BANA: signal strength Auth.	[132]	X				X	X			X
25	In-vivo NFC	[133]	X	X		X				X	X
26	Heart-to-Heart ECG Proximity	[80]	X	X					X		X
27	Hospital Auth. Server AC	[134]	X	X		X	X	X		X	
28	Secure Touch Screen for WMDs	[135]	X			X		X		X	X
29	Trusted platform module (TPM)	[111]	X	X			X	X	X		X
30	Differential Power Analysis counters	[111]	X			X				X	X
31	Run-time Monitoring for Sec. Exec.	[136]	X	X	X	X				X	
32	Heisenbyte: Memory Disclosure	[137]	X						X		X
33	Software Symbiotes	[138]	X	X	X	X			X	X	X
34	WattsUpDoc: Sidechannel monitor	[139]		X	X	X				X	X
35	Blockchain-based firmware update	[140]		X	X						X
36	Repairable control logic	[141]		X	X	X					X

Table 2: Security Mechanisms Synthesized from [2, 56, 79, 57, 135, 111, 136, 81, 137, 138, 114, 139, 140, 142, 141]. Initial structure of this table came from [2].

.

and authentication. Halperin et al. [113] and Ellouze et al.[114], attempted to use biological systems to power communication/security functionality instead of tapping into pacemaker and other IMD batteries. The intention of this mechanism is to prevent battery/resource consumption threats.

3. **External Cloaker** is an approach proposed by Denning et al. [115] that uses an external device to mediate interactions from PMDs and their programmers or other connected components in the LCSs they reside within.

4. **Proximity-Based Access Control** is a mechanism proposed by authors such as [116, 56] that limits access according to a defined distance metric. Access attempts at a distance exceeding the threshold are denied to force attackers to be very close to the PMD to carry out attacks.

5. **UV Ink Tattoo as a Key** is an approach similar to wearable wristband passwords that apply a UV ink as a tattoo on the patient's body. The ink acts as a password that limits access to a PMD. Denning et al. [118] and Schechter et al. [117] propose similar approaches.

6. **Integrated Communication with a Cell Phone** refers to work by Hei et al. [119] that demonstrates a proof of concept to address resource depletion attacks that attempt to overflow a device with large amounts of data.

7. **Subcutaneous Button** is an approach proposed by Hansen and Hansen [6] in the taxonomy of vulnerabilities already summarized in our Section 2.2.4. The idea is to embed a device in an IMD that, when pushed, allows incoming communication. When not pushed, the IMD would reject all traffic. This approach is purely conceptual.

8. **IMD Guard** is a mechanism propsed by Xu. et al. [120] that is similar to the external cloaker. In this case, IMD Guard uses ECG-based keys and an external device to mediate access control to a PMD.

9. **Rolling Code** refers to a mechanism proposed by Li et al. [121] that is similar to a garage door opener. The idea is to use a shared key and a sequence counter that ticks up with every access of a PMD.

10. **IMD Shield** has been proposed in a few places, including [122, 79]. The idea is similar to External Cloaker and IMD Guard in that it uses an external device to mediate access.

11. **Lightweight Security Protocol** is a family of approaches that seek to use only low-energy resources to implement authentication and encryption. Housseini et al. are one such approach [123].

12. **One-Time Pad Encryption System** is an approach proposed by Kaadan et al. [124] that uses a one-time pad encryption technique for PMDs to make cipher text impossible to decrpyt without the key.

13. **Personal Security Device** is an external device created by Pournaghshband et al. [125] that mediates interactions between a PMD and a programming device elsewhere in the LCS.

14. **Biometric Identifiers** refer to a family of different approaches such as [2, 56] that make use of life-based systems, such as heartbeat, to generate keys or other inputs for encryption.

15. **Ghost Talk: Analog Defenses** is a system proposed by Kune et al. [126] that seeks to mitigate electromagnetic interference signal injection attacks by augmenting sensors with attenuation capabilities.

16. **Split-core System Architecture** is an approach proposed by Strydis et al. [127] that separates operational LCS resources from communication resources to prevent battery or other resource depletion attacks.

17. **Anomaly Detection Using Bowel Sounds** is an interesting idea by Henry et al. [128] to detect anomalous usage patterns, particularly in insulin pumps. The idea is based on the notion that diabetic patients inject insulin after eating. With this knowledge, insulin application at times when the bowels are less noisy may indicate an unwanted dosage.

18. **MedMon** by Zhang et al. [129] is similar to other external devices like External Cloaker, IMD Shield, and IMD Guard. Like the others it acts as a mediator. Unlike the others it is more focused on anomaly detection and utilizes alerting which enables prevention mechanisms to make odd or potentially malicious behaviors known to the patient. [129]

19. **Proxy Device** is shorthand for a device proposed by Darji et al. [130] to detect active attacks on wirelessly capable IMDs. It is similar to other external mediators.

20. **Infusion Pattern-Based AC** is an implementation of access control by Hei et al. [131] that prevents overdoses of insulin by detecting anomalous insulin consumption dosages.

21. **Physical Layer Authentication** is similar to proximity-based access control, but uses a "pre-equalization" technique [2] to estimate the distance between a PMD and an external device using communication channel physics. External devices outside the LCS are denied access if they do not meet a threshold.

22. **Emergency Access Overrides** is not a security mechanism per se, but is closely related. The idea, best summarized by Camara et al. [56] emphasizes the tradeoffs between security mechanisms and the importance of timely accessibility to PMDs for emergency/first responders. The idea suggests that any security mechanism will need intelligent overrides that balance security against safety during crisis.

23. **Wristband Passwords** are a class of devices often paired with PMDs for LCSs. Works such as [56, 79] speak to them. Wristband passwords are basically tokens in the sense of being *something you uniquely have* and it can be easily associated with LCSs to act as an authenticating identifier. The problem with this approach, like the UV ink tattoo, is that attackers can sniff the password and replay it.

24. **BANA: Authentication Using Received Signal Strength Variation** is an approach proposed by Shi et al. [132] that examines channel strength in body area networks and builds patterns related to how signal strength varies. It is similar to other proximity-based approaches but better couched in body area networks as opposed to a strictly mediating role.

25. **In-vivo NFC** is a mechanism explored by Kim et al. [133] for using NFC-based technologies commonly equipped on cell phones and other devices to remotely monitor PMDs. The idea is to embed an NFC-capable component attached to the IMD or WMD and use it for pairing to a smart phone, potentially as a gateway for other web/cloud components in the LCS.

26. **Heart-to-Heart ECG Proximity** is an innovative use of heartrate as an authenticator. As Rostami et al. describe [80], an ECG is used to measure heartrate by a PMD. Any device wishing to interact with the PMD, must also collect and exchange the same ECG signal for some duration to facilitate communication. Rostami's work provides a cryptographic device that uses the ECG signal to generate a sufficiently random key to thwart adversaries seeking to bypass the authentication mechanism.

27. **Hospital Auth. Server Access Control** is an authentication mechanism proposed by Park et al. [134] that provides PMDs with a long-term key to pair with a hospital server. Physicians also receive a similar key. By combining the PMD and physician key, access can be granted to the PMD for incapacitated patients.

28. **Secure Touch Screen for WMDs** is the most salient feature in a patent taken out by Debelser et al. for augmenting infusion pumps [135]. The idea is that the screen can be augmented with security features to prevent unauthorized access by checking security data such as password, biometrics, etc. associated with the user access attempt.

29. **Trusted Platform Module (TPM)** is a security feature often used in IoT and wearable devices. Summarized well by Al-Omary et al. [111], a TPM is basically an embedded chip that can co-exist with other electronics. The chip uses hardware acceleration to hold an RSA key pair which is used cryptographically prevent component modification and ensure secure booting. While useful in WMDs, power consumption can be an issue for IMDs.

30. **Differential Power Analysis Countermeasures** seek to prevent attackers from observing chip/board behaviors and using observational data to detect encryption keys. Would-be attackers use oscilloscopes and probes to attach to a chip as it operates. Countermeasures attempt to prevent this by adding noise and other probe-prevention capabilities to devices. The idea is summarized well in [111].

31. **Run-time Monitoring for Secure Program Execution on Embedded Processors** is an implementation created by Arora et al. [136] to ensure that programs do not deviate from their intended permissible behaviors. Their approach uses a hardware monitor and invariants to check procedural flow control and instruction integrity at the processor level.

32. **Heisenbyte** is a system created by Tang et al. [137] to prevent memory disclosure attacks by using destructive code that garbles code right after it is read into memory. The approach works for both static and dynamically generated code and has relatively modest overheads according to experiments conducted by the authors.

33. **Software Symbiotes** is an approach demonstrated by [138] that injects functionality into device firmware to add intrusion detection and prevention functionality, even for devices that don not have existing capabilities. The symbiote code is injected in a way that allows it to stealthily protect itself from removal. The authors demonstrate a rootkit detector Symbiote that was injected into a CISCO IOS-based device. It is unclear from the work if the approach could be suitable for LCSs or not, but it is highly integrable with legacy embedded devices.

34. **WattsUpDoc: Sidechannel Monitor** is a monitoring system created by Clark et al. [139] that observes behaviors in power consumption. Modeling power consumption as a side-channel and proxy of normalcy, WattsUpDoc was purportedly able to detect previously known malware with 94% accuracy and previously unknown malware with 85% accuracy. The system has a low overhead and is designed to be used on embedded medical devices.

35. **Blockchain-based Firmware Update** is a technique created by Hu et al. [140] that makes use of blockchain to create a smart contract and ensure that firmware updates are malware proof. It works by verifying a checksum and download URL against the blockchain.

36. **Repairable Control Logic** is a general class of resilient coding techniques that provides protections against design flaws. Wagner et al. [141] propose a hardware patching mechanism that detects design errors as they manifest in erroneous program execution states. Their "state matcher" switches a device's processor into a "degraded performance mode" that allows the device to correct the error. Once corrected, the device is switched back into normal execution mode. These kinds of approaches are desirable for LCSs, particularly for use during patching.

This listing of security approaches is not meant to be exhaustive, but it does provide a purposeful sampling of a wide range of innovations to defend LCSs from malicious attacks. These mechanisms range from conceptual ideas to commercial offerings in their technical readiness. These security mechanisms also apply at different levels in the embedded system stack. Finally, the diversity of mechanisms and lack of rigorous field testing points to the fact that medical device security is still evolving and adoption in commercial LCS offerings is minimal.

Distilled literature finding(s):

- L16: Medical device security is still evolving and security mechanism adoption in commercial LCS is minimal.

2.3.5 Existing Validation Approaches

In the late nineties, FDA found that 79% of software-related recalls of medical devices within a five year span were due to software defects introduced when changes were made to the software after its initial production and patching [1]. More recently, Kuehn et al. [60] reported similar issues in a patch released for the Abbott pacemaker. To address these issues, FDA regulatory pressures have motivated the development of methods to assess compliance, measure and rank device features, or assess and manage risk.

It is important to recognize that devices that constitute LCS are not a single embedded device, which naturally complicates testing. Modern medical devices span three domains: embedded systems, mobile devices, and web services. To consider security challenges in each domain separately, Hale et al. [143] have developed the SecuWear platform. This platform allows domain appropriate skill sets and tools to be brought to bear. While platforms like SecuWear are great for post-market analysis, the patient-machine interface often receives greater attention in Premarket analysis. Patient-machine interface has direct implications for patient safety due to usage ambiguity or device malfunction. Validation of requirements for process steps, timing, and design features are notably grounded in formal modeling and reasoning [144, 145, 146]. Another major FDA compliance focus is on software development practices i.e. processes are planned for and followed when developing and maintaining software for medical devices. To target this need, Lepmets et al. [147] have developed a framework called MDevSPICE that can be used to assess readiness for regulatory audits of software development practices. Similarly, manufacturers can also use this framework to assess the practices of their software suppliers.

In the absence of specific recommendations, the general FDA recommendations have led to many formal methods for modeling, design, development, verification, and validation of software-intensive medical systems [148]. Bonfanti et al. [148] have observed that since 2010 there has been a significant uptick in formal and rigorous methods proposals in support of functional safety, cybersecurity, and reliability for medical devices. Software validation using formal methods examines the alignment and deviations of actual software behavior with respect to the model of the specification. Hence, it is desirable that formal methods integrate with the software development life-cycle and seamlessly connect a high-level specification of the system to implementation level details (e.g. code) [145]. Such formal methods are concerned with establishing the adequacy of control (functionality and interactivity) provided by software-intensive components from a medical/physical process, human-machine interface, and timing perspective [149, 150, 144]. These perspectives also align readily with recommended standards for software life-cycle processes and related tasks for validation. However, there is a lack of consideration of a malicious adversary tools, techniques and practices (TTPs) in formal models focused on safety and reliability. In the absence of a threat model, validation assurances lack arguments and evidence for cybersecurity. This omission is acknowledged by the recent inclusion of threat modeling activities in the draft FDA Premarket guidance for cybersecurity [39]. In combination with threat modeling at the design level, programming language guidelines that call attention to insecure coding practices (e.g. use of MISRA C for a hemodialysis machine [151] and SW91[108] defect taxonomy) are necessary for validating a user's cybersecurity needs and expectations being met by the medical device.

The formal notations typically used in literature are skewed towards state-based notations such as Automata, Event-B, Z, and Extended Finite State Machine (EFSM) [148]. These notations are popular for a systematic refinement of requirements models into functional code to demonstrate design validation in compliance with QS regulations. Bonfanti et al. have also found that "*the most common activities performed are modeling and model verification, followed by model validation. A trend towards code generation has also been observed in recent publications. Although activities like software validation show little traction*"[148]. While there is no shortage of formal and rigorous methods along with tool support, these approaches are only recommended (hence, not required to be implemented) by regulation or FDA approved consensus guidelines [37]. As a result, software validation and, by extension, patch validation research is limited.

Finally, we provide a survey of firmware analysis tools that offer capabilities for software validation in Appendix A.

2.3.5.1 Automated Patch Generation and Validation

Typical vulnerability discovery and patch cycles involve a great deal of time and effort from developers and testers. Manufacturer-issued patches have long lead times and are reactive in that they can only respond to known, fixed attacks. The goal of automated patch generation and validation is to correct defects in production systems at run-time without manual intervention [152]. Patch generation is best viewed as a search problem: First, limit the search space based on a finite set of transformations and then locate correct patches within this search space. While the process is simple, establishing patch correctness is a hard problem. Qi et al. [153] have also found that many automatic *generate-and-validate* systems produce incorrect patches that may introduce new security vulnerabilities or eliminate intended functionality. Their analysis shows that most patches generated by current state-of-the-art automated systems are incorrect, i.e. they do not produce a correct output for all items in the input test suite. Among those patches that are correct, a large number of them are semantically equivalent to simply deleting the defective functionality. This suggests an interesting avenue of future research, where automated patching results in a degraded mode of operation that limits attacker access while still preserving critical functions. More recently, Xiong et al. [154] report much success in using similarity in program execution to determine patch correctness. Using just two program execution heuristics, their approach was able to successfully filter out 56.3% of incorrect patches from state-of-the-art automated patch generation systems without losing any of the correct patches.

Patch generation systems have also found applications in autonomous systems capable of playing in a capture-the-flag hacking competition [106]. In the DARPA cyber grand challenge, a set of challenge binaries are available to the participants. The challenge binaries implement a network connected service. The goal is to patch the challenge binaries to neutralize attacks while also preserving functionality and minimizing computational overhead. Patching can be in the form of a direct modification to binaries or rule-set updates for an Intrusion Detection System (IDS). The attacker is assumed to have access to the modified binaries as well the IDS rule-sets, which are valid assumptions to make for commercially available medical devices. A participant system in the DARPA cyber grand challenge called Xandra won the security score and $1M prize with

an approach that combined fuzzing strategies with input selection based on symbolic execution to achieve a greater program execution coverage. This strategy allows fuzzing to quickly find inputs that result in program crashes, an indicator of software bugs and vulnerabilities. They also used Zipr [155], a static binary rewriting tool that builds a compact and efficient transformed binary program or library to patch security vulnerabilities. Zipr allows user-specified transformations to modify an existing binary program to improve its security. Zipr provides an API that allows users to browse, change, or remove instructions [155]. Finally, the DARPA Cyber Grand Challenge was very significant for pushing the bounds of existing software analysis techniques; however the participating approaches were very specific to a single computer architecture and performs a limited form of software validation on the patched binary using a set of pre-defined input sequences to collect measures of execution time, file size, run-time memory usage, and functionality.

Compared to x86 binaries, it should be noted that medical device software and firmware pose an additional level of difficulty for patch validation. Dynamic testing of embedded device firmware binaries requires a high fidelity model of all special-purpose hardware components that it uses in the real system. To overcome this difficulty, Zaddach et al. [107] have developed a framework to execute firmware code on an emulator, which forwards all memory and peripheral I/O access requests to the real device under test. Extending such tool-chains to provide evidence for re-certification of patched embedded system software is a promising area for future research.

Automated patch generation has also enabled deception capabilities. Deception here can take the form of faux, obfuscated, or active response patches mixed with legitimate ones [156]. Deception capabilities here are aimed to waste time and mislead attacker techniques that analyze device patches to see what bugs and vulnerabilities have been fixed, so that they can efficiently develop and weaponize exploits to target unpatched devices.

Distilled literature finding(s):

- L17: Many automatic *generate-and-validate* systems produce incorrect patches that may introduce new security vulnerabilities or eliminate intended functionality [153].
- L18: Use of formal methods has proliferated in the medical device manufacturing community, but their applications are primarily for proving safety and medical process effectiveness, not cybersecurity. This research area is highly lacking for high-confidence software and patch validation for cybersecurity.

2.3.6 Existing Verification Approaches

According to the QS regulation, verification means *confirmation by examination and provision of objective evidence that specified requirements have been fulfilled.* In terms of design, verification requires confirming that the outputs of the design process such as code meet the design inputs and requirements. In any case, verification requires the creation of models at appropriate levels of abstraction to prove properties established in the specification. Given the expected rigor in verification, Allen [157] highlights the delicate balance between regulatory pressures and freedoms that

enable timely innovations in the medical device industry. While acknowledging growing complexity of medical devices, he discusses the large improvements in static and dynamic analysis tools for software that are being used by device manufacturers to identify coding defects early in the development life-cycle. While functional software properties can be proven with absolute guarantee, the same is not true about proving the absence of coding defects. The risk of coding defects is minimized to an acceptable level using a combination of code review and tool-based scanning results. Additionally research concepts presented in Section 6.1 can reduce risks of vulnerabilities not caught by these techniques.

Among medical device case studies that apply formal methods, the most common step after modeling activities is model verification [148]. Bonfanti et al. [148] have found that model checking based verification techniques are applied quite frequently in prior work. Model checking verifies that a finite-state system model satisfies critical system properties specified in temporal logic. Other verification techniques include Satisfiability Modulo Theories solvers (SMT) and Theorem provers. While formal methods have their applications, malicious actors sometimes chain together actions that were not forseen by the original designers. Therefore, in the verification of medical device security mechanisms there should be attempts to bypass defenses by a team of technically competent individuals that are not part of the original designers or developers.

Developing models for complex biomedical systems can be a significant challenge. For complex cyber-physical systems there might be multiple levels of system abstractions involving different verification paradigms and tools. In such cases, it is important to reconcile abstraction induced differences among different verification artifacts. Murugesan et al. [158] demonstrate the use of notations and tools appropriate for verification tasks related to architectural model, behavioral model, and system timing properties for a patient controlled analgesia infusion pump and then compose results for manual review.

Model-Based Engineering (MBE) approaches face similar model integration challenges with different models representations for device, architecture, information, software, control, domains, behaviour and other views [159]. These models need to be integrated for component and system level verification assurances. MBE recommends model transformations and mapping to address these issues. However, integration of cross-cutting constraints like security across different model views remains an open challenge in MBE. If cross-cutting constraints are not integrated as first-class elements into the models, then impact on the system cannot be fully understood upon changes and updates. Finally, in such cases it is worth considering segmenting multi-level systems and implementing risk controls presented in Section 6.1 for each segment to safeguard the interaction between critical components.

Finally, there is also a growing interest in including medical devices in cybersecurity challenges and contests (e.g. the DEFCON BioHacking Village [160]) to raise awareness of cybersecurity issues. Although informal in nature, challenges and contests do provide weak assurances for medical devices being able to withstand a malicious and dedicated adversary attempting to bypass normal security measures.

3 Clinical Perspectives

Foundational studies such as the Halperin et. al.'s 2008 pacemaker hacking study [161] and a 2011 Black Hat presentation about Medtronic insulin pump hacking [162] drove much of the early attention to potential dangers in this field from the medical community and public. Recently there are anecdotal reports of similar dangers manifesting on newer LCS such as hacking incidents directly impacting the brain surgery of a child in Russia [163, 164].

Clinical stakeholders and patients are sometimes very aware of the baseline injury and death rates caused by medical devices, as "across all types of medical devices, more than 1.7 million injuries and nearly 83,000 deaths were reported to the FDA over the last decade" [42]. Given these statistics and the technology challenges discussed thusfar, we posit that it is difficult to distinguish between normal or cyberattack caused injuries or deaths to patients. This theory is supported by reports of glitches in devices that may appear like cyberattacks to users [43].

Additionally, recent clinical simulations on scenarios with biomedical device cyberattacks have demonstrated supporting evidence for this theory. In these scenarios, clinical professionals are significantly challenged by cyberattacks against LCS and the IT systems they depend upon for electronic medical records. In one simulation a patient arrived in the ER in cardiac distress, with his ICD shocking him every minute due to a cyberattack. The clinical staff struggle to diagnose the situation and eventually resort to cutting the pacemaker out of the patient [165]. Another live simulation featured an ER patient who was the victim of a cyberattack targeting the hospital's infusion pumps. The scenario ends when the physician being tested asks to replace the infusion pump, but at that stage the patient had potentially fatally overdosed on IV medication [166]. These scenarios, hosted by the CyberMed Summit, involve unsuspecting medical doctors in live simulations involving biomedical devices and healthcare IT systems under cyberattack [167]. In general the simulations find that the medical staff at most consider that the systems are glitching, being buggy, or "running away", but do not communicate suspicion of cyberattacks at play. To gain a more complete understanding, we present an IRB-approved survey of various LCS stakeholder perspectives.

Distilled clinical finding(s):

- C1: The difficulty of detecting cyberattacks is a profound concern for clinical stakeholders and patients who would be on the front line of a biomedical device cyberattack.

3.1 IRB-Approved Clinical Survey of Stakeholders

Our medical team conducted an anonymous survey of diabetes-related LCS patients, doctors, and medical device software developers to provide quantitative results on the clinical perspective to measure the scale and magnitude of basic security and patching challenges for LCS. Our focus was to obtain high quality data, and we deliberately avoided constructing any question phrasing that could influence or lead respondent answers. We obtained IRB approval for the surveys, and conducted them with informed consent. The survey was conducted over 4 months online, in clinics, and at the DEFCON 2019 BioHacking Village, and received the following respondent totals:

- 57 diabetes-related LCS Doctors. These are respondents that prescribe diabetes-related life critical systems to their patients.
- 61 diabetes-related LCS patients. These are patients who utilize diabetes-related life-critical systems in their Type 1 or Type 2 diabetes treatment.
- 11 diabetes-related LCS Developers. These are device developers in the medical device industry who design, develop, or test diabetes-related life critical systems.

3.1.1 Survey Results from Patients and Doctors

1. C2: 62% of diabetes-related LCS patients said that increased costs with new devices can prevent them from upgrading device models. This indicates that it is not sufficient to address security alerts and vulnerabilities by forcing users to purchase upgrades.
2. C3: 46% of diabetes-related LCS patients are unaware of whether their device can be updated or patched. This indicates that there is an awareness gap in LCS device documentation or training.
3. C4: 72% of patients that do patch their diabetes-related LCS demonstrate caution or distrust of the patch. This indicates that there is a need for patch assurances.
    - 38% of diabetes-related LCS patients apply a patch only when the current software stops working;
    - 34% of diabetes-related LCS patients apply a patch only **after** they read what has been changed;
    - 23% apply the patch as soon as available or as soon as they get home;
    - 5% do not pay attention to device updates.
4. C5: 20% of diabetes-related LCS patients interact with or modify their devices through do it yourself (DIY) methods. 58% of those respondents report to be following the #wearenotwaiting effort. This indicates there is a significant patient demographic that relies on modifying their own LCS.

5. C6: There are multiple responses that indicate there is a broad cybersecurity assurance gap in diabetes-related LCS:
    - 90% of diabetes-related LCS patients report that there are no cybersecurity assurances for their device provided to them.
    - 65% of diabetes-related LCS doctors have concerns about the confidentiality, integrity, and availability of the devices they prescribe.
    - 59% of diabetes-related LCS patients have concerns about the confidentiality, integrity, and availability of data on their device(s).
    - 25% of diabetes-related LCS doctors report that their patients have also expressed concerns about the security of diabetes-related LCS.
    - 59% of diabetes-related LCS patients feel that even if a device is cleared by the FDA, it does not deliver sound cybersecurity.
6. C7: There are multiple responses that indicate there are patch validation gaps for diabetes-related LCS:
    - 36% of diabetes-related LCS patients report integrity and availability impacts caused by updating the mobile phone operating system software (e.g. Android/iOS) when they use it to support their continuous glucose monitor.
    - 31% of diabetes-related LCS patients report integrity and availability impacts caused by updating the continuous glucose monitor's mobile application.
    - 62% of diabetes-related LCS patients already do not always trust the data provided by their medical device, either because of availability or integrity issues. We believe this is primarily due to the existing inaccuracies in glucose measurement technologies and approaches.
7. C8: 100% of diabetes-related LCS patients report strong physical security practices for their device use.

3.1.2 Survey Results from Medical Device Industry

The following findings are conclusions we present from analyzing the anonymous responses from the medical device industry. They do not represent the opinions or practices of any one manufacturer.

- I1: 50% of the industry respondents are unaware of DIY efforts.
- I2: 50% of respondents report they have observed availability or integrity impacts with continuous glucose monitors from either phone operating system updates or mobile application updates. Some medical device manufacturers even track devices and equipment that are known to cause interference with their products[6].

[6] http://www.medtronicdiabetes.com/customer-support/equipment-interference

- I3: 50% are concerned with the existing accuracy of their devices. This may be because the state of glucose measurement is an imperfect science.
- I4: 90% of respondents report they do think about the security of their devices.
- I5: 80% of respondents report they disable JTAG and other programming interfaces on the device once it is ready for the consumer.
- I6: 50% of respondents report they allow debugging of medical devices post-production.
- I7: 90% of respondents report that they intend for devices to be updated after they have been manufactured.

4 Exploratory Research and Development

We performed exploratory research and development on two proof of concept technologies, utilizing short term Agile sprints. Exploratory research sprints were compressed into 30 days each and strictly focused on testing feasibility of concepts that could enhance the security or patching challenges for LCS. Each sprint's objectives was to establish a working proof of concept and determine both viability and challenges for future work. Several sprints continue beyond the writing of this study and are documented in separate technical reports. We explored two proof of concept technologies that could simplify the challenges in LCS security and testing:

1. A novel approach to mitigating the impacts to LCS, termed a Bio-Firewall; and
2. A novel I/O centric approach for dynamic analysis, exploring the limits of open source emulation and fuzzing to enable scalable testing platforms for embedded systems.

We share the details and discoveries of our exploratory research in Sections 4.1 and 4.2. Additionally our exploratory research involved device teardowns. The raw results from these device teardowns are contained in a separate technical report and summarized findings are available in Section 4.3.

4.1 Proof of Concept R&D - Bio-Firewall

[redacted]
L4.
[redacted]
7, [redacted]

4.1.1 Our Approach

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

- E1: [redacted]
- E2: [redacted]
- E3: [redacted]
- E4: [redacted]
- E5: [redacted]

4.2 Proof of Concept R&D – Exploring Automated I/O fuzzing of a Virtualized LCS

The research questions we explored in this sprint were as follows:

- To address the challenges of dynamic analysis testing of LCS devices per L13: If firmware from LCS components could be tested in a high fidelity hardware emulation platform, could the system inputs and outputs be automatically targeted for fuzzing to offer novel virtualization capabilities for blackbox testing?

- To address the challenges of software supply chain complexity and access restrictions, especially for legacy systems per L12: Could this address the majority of legacy LCS that have software components that no longer have available source code for testing?

4.2.1 Our Approach

This sprint's objective was to explore the feasibility of automating I/O fuzzing once a LCS was virtualized via high-fidelity emulation.

Our target was the nRF51-DK experimental continuous glucose monitoring application. The nRF51-DK was chosen for this proof of concept because its controller is in the same family as that of the Dexcom G6 CGM. Additionally, the nRF51 software development kit (SDK) provides precompiled and linked wireless protocol stacks called "softdevices" that provide an interface between the application and the Bluetooth Low Energy (BLE) features of the microcontroller. The SDK also contains example source code for various applications including simple BLE transmission, fitness tracking, and medical devices. One example application is named "Experimental: Continuous Glucose Monitoring Application" and implements simple CGM functionality that takes glucose measurements every 60 seconds and sends status updates to a connected central device (smartphone, tablet, etc.). Due to the hardware and processor commonalities, we have assumed that the Dexcom G6 functionality is implemented similarly to this example application, making it a reasonable analogue on which to build a proof of concept.

We utilized a coverage based fuzzing and emulation platform called AFL-Unicorn[7], which is based on QEMU[8]. AFL[9] is a powerful tool for coverage-based fuzzing in open source applications [105] . This tool was combined with the Unicorn emulation[10] engine to establish "AFL-Unicorn" by Battelle to "*let you fuzz any piece of binary that can be emulated by Unicorn Engine*" with the AFL coverage-based fuzzing engine. AFL, QEMU and Unicorn Engine are discussed further in Appendix A.1.

Investigation

In order to provide this type of virtual fuzzing, some static analysis must be performed to produce a valid starting machine state and identify input buffers. Unfortunately, QEMU is written with only a single function to trigger guest interrupts to the emulator, which must be called by an outside process to QEMU itself. One must develop this separate I/O handler program to call

[7] `https://github.com/Battelle/afl-unicorn`
[8] `https://www.qemu.org/`
[9] `http://lcamtuf.coredump.cx/afl/`
[10] `https://www.unicorn-engine.org/`

this emulation interrupt function in QEMU. This is a significant challenge which is glossed over in the QEMU documentation. We began development of this separate program to handle the I/O interrupt trigger of the QEMU emulator in Python by leveraging its flexible bindings and memory handling.

Experimental I/O Handler Research & Development

In our experiments, we used NSA's Ghidra coupled with nRF51-DK and ARM Cortex M0 documentation to perform static analysis to identify the necessary input buffers for BLE communications and a valid starting machine state taken from a development board. After isolating a state as close to the beginning of the BLE advertising driver function as possible, we created a program in Python which would write the contents of a Bluetooth packet capture file to the input buffer and set the interrupt flag that a packet was received. Unexpectedly, when running the system here, QEMU simply ignored that the interrupt flag was set and continued as though no packet was received, which revealed we needed to develop a more accurate simulation of the peripheral behavior on top of the interrupt project.

However, the development of a sufficiently accurate simulation of each peripheral and modifying the current Python AFL-Unicorn implementation to accomplish this objective proved to take a degree of significant dedicated development effort that was beyond the scope of this research. The approach is definitely feasible; but was just out of scope for this study which lead us to pause this project. This was a discovery and other worthwhile findings were made after the implementation was nearly completed in Python.

Our research and development efforts did find avenues for addressing the performance challenges in developing a high-fidelity emulation, such as:

- Other research in the field which leverages Avatar2 from Python to orchestrate AFL-Unicorn with several other tools suggests migrating to Pypy Just-In-Time (JIT) compiler versus Cpy's precompiler for significant performance boosts [168].
- Additionally, dedicated orchestration frameworks such as DeepState could create a significantly more performant system than current Avatar2-driven project benchmarks which suffer from the performance overhead of Python 2 [168].

Distilled exploratory research and development finding(s):

- E6: We found that the currently embedded system virtualization available tools still require a large development investment for each available target while lacking fidelity and run-time efficiency. Specifically, Our research on the tools identified unwanted degradation in performance as compared to the physical device when research operated in hybrid emulation mode and no increase in performance when full emulation mode was compared to the physical device [168, 169].

- E7: We identified the need for mature, feature-rich orchestration frameworks to support virtualization of LCS components and their I/O.
- E8: We identified that the NRF51 DevKit is valid surrogate model development kit to use for further research relating to continuous glucose monitors.
- E9: We determined that fuzzing virtualized LCS is feasible given a larger period of time dedicated to this goal.

4.3 Device Teardown Summary

We summarize sanitized findings from our teardown efforts here. Our full teardown report is available in a separate document. Vendors are anonymous for this discussion. Our approach procured the top CGMs and Pacemakers recommended to us by clinical doctors, and involved hardware analysis and destructive methods to determine the following information:

4.3.1 Continuous Glucose Monitor Teardowns

Below are our findings from our CGM teardowns.

- T1: We identified architectures used in this environment, such as:
  - Vendor 1 has evolved primarily over the NRF 51 and NRF52 series of processors.
  - Components such as smartphones, tablets, and dedicated receivers commonly have other ARM processor variants.
  - Vendor 2 device series seems to use the msp430 processor.
- T2: We encountered a steep level of hardware security measures which also prevent updates.
  - The casing for the Vendor 1 device is designed so that the outer casing and the battery are close to the transmitter which has small wings. The wings secure the transmitter down inside the sensor. This setup makes it difficult to grind down the exterior shell.
  - The battery connector terminal is much smaller and thinner than previous Vendor 1 device versions. This changes makes it difficult to remove the battery without damaging its terminals.
  - Once the casing was removed from the transmitter, it was determined that the silicone dies were obfuscated with a stiff potting compound rendering identification incredibly challenging.
  - The Vendor 2 device series utilizes the msp430 microprocessor security bit to disable processor introspection via JTAG.
- T3: The dedicated CGM receivers and transceivers are not intended to be updated or patched and do not have a working debug port such as JTAG.
- T4: The associated smartphone applications are all updated through the respective device's application store.

4.3.2 Pacemaker Teardowns

- T5: The pacemaker programmers do not utilize full disk encryption.
- T6: The pacemakers themselves receive firmware updates from the programmers when necessary.
- T7: The pacemaker programmers receive occasional updates from the manufacturer via signed usb drive.
- T8: The pacemakers' PCBs often have internally exposed test points, which are available avenues for future proof of concept R&D efforts.
- T9: The programmers generally leave functional debugging interfaces, which are available avenues for future proof of concept R&D efforts.
- T10: The only processor commonalities we have identified so far in our LCS teardowns and in existing firmware analysis tools (Appendix A.1) is ARM. Future research for software and firmware analysis would benefit from identifying what existing analysis capabilities for ARM can be extended to LCS. However, we have not exhaustively analyzed a broad amount of LCS devices, just the most popular CGMs and pacemakers.

5 Research Findings Summary Tables

This section collects all of our distilled findings from Sections 2-4 into multiple reference tables. A link to the subsection where each finding is discussed is provided.

5.1 Literature Review Findings

Finding #	Literature Review Finding Text	Section
L1	It is important to recognize that LCS and biomedical systems are often not a single embedded device. Consequently, a comprehensive framework for securing and patching LCS should consider embedded systems, mobile devices, and web services.	2.1
L2	Manufacturers are responsible for making an argument and collecting evidence for demonstrating the safety and effectiveness of their devices. However, self-examination is a poorly regarded assurance mechanism for security. FDA certified 3rd-party device certification labs could provide higher levels of assurance in Premarket security design and post-market patch validation.	2.1.2.1
L3	The fact that most medical device vulnerabilities require low skill to exploit indicates there is a gap in vulnerability testing, threat modeling, or security controls of software and patches in medical devices.	2.2.1
L4	It is clear that LCS present a large attack surface with many possible threat vectors. Unlike industrial systems with many assets and actuation points, the impact surface of an LCS that can cause patient harm is often limited and focused. These focused biological processes should be monitored for anomalies (e.g. IDS/IPS) during normal operation and while patching.	2.2.3.1
L5	For an LCS, internal-devices are low-powered and access requires physical proximity, but internal-LCS and external components are exposed to network and cloud-based infrastructures. Thus when securing and patching internal-devices, these devices should not have any inherent trust in other LCS components.	2.2.3.1

Finding #	Literature Review Finding Text	Section
L6	Malicious actors sometimes chain together actions that were not foreseen by the original designers. Therefore, in the verification of medical devices security mechanisms there should be attempts to bypass defenses by a team of technically-competent individuals (e.g. 3rd-party FDA certified assessment labs) that are not part of the original design or development teams.	2.2.3.1
L7	Transparency of hardware, firmware and software components is an important need for patch and configuration management, and this gap inhibits healthcare organizations' efforts to manage cybersecurity risks to biomedical systems. This may be addressed by the cybersecurity-bill-of-materials proposed by the draft Premarket guidance [39].	2.2.4
L8	Systems that make up LCS and biomedical systems often use vulnerable legacy operating systems, software, and components which should be considered in comprehensive security and patching strategies. Compromised legacy systems can act as easy gateways for other attack vectors to threaten other biomedical and LCS targets.	2.2.4
L9	Medical device manufacturer assessment and patching of 3rd party components in LCS is a ubiquitous, significant gap despite longstanding FDA requirements to do so. Many vendors have historically offered LCS with known unpatched vulnerabilities in the 3rd party components.	2.2.4
L10	Reproducibility of bug and vulnerability reports as well as threat evaluation findings is a challenge for LCS [79] due to a lack of access to devices for 3rd party security researchers and the velocity of LCS change in the market.	2.2.4

Finding #	Literature Review Finding Text	Section
L11	To facilitate risk management efforts, new scoring methods are needed to evaluate the complexity of assuring correct operation of an LCS patch that meets all user's needs and intended uses. Such a scoring system should be able to accommodate the entire embedded system stack including firmware and hardware and all input/output interfaces, such as network, sensors, and actuators for effective estimation of patch validation complexity. Such a scoring system should also accommodate the assurance needs of LCS end-uses and care-providers for patches.	2.2.5
L12	Given complex software supply chains and intellectual property rights, manufacturers and 3rd-party certifiers alike may never have full access to software source code. A combination of white, grey, and black box testing is required for different software levels in an embedded system.	2.3.1
L13	Dynamic testing of embedded device firmware binaries will require a high fidelity model of all special-purpose hardware components that they use in the real system. Emulators that support hardware in the loop operation are needed to accommodate diverse or custom hardware components used in LCS. Such test-beds are essential to further the science of securing and patching LCS.	2.3.1
L14	Diverse microarchitectures, instruction sets, and customized input/output (I/O) channels hinder high-fidelity virtualized simulations for embedded system operations [107].	2.3.1
L15	There are few available firmware analysis tools that can be applied to LCS architectures (discussed further in Appendix A)	2.3.1
L16	Medical device security is still evolving and security mechanism adoption in commercial LCS is minimal.	2.3.4
L17	Many automatic generate-and-validate systems produce incorrect patches that may introduce new security vulnerabilities or eliminate intended functionality [153].	2.3.5.1

Finding #	Literature Review Finding Text	Section
L18	Use of formal methods has proliferated in the medical device manufacturing community, but their applications are primarily for proving safety and medical process effectiveness rather than cybersecurity. This research area is highly lacking for high-confidence software and patch validation for cybersecurity.	2.3.5.1

5.2 Clinical Survey Findings

Finding #	Clinical Finding Text	Section
C1	The difficulty of detecting cyberattacks is a profound concern for clinical stakeholders and patients who would be on the front line of a biomedical device cyberattack.	3
C2	62% of diabetes-related LCS patients said that increased costs with new devices can prevent them from upgrading device models. This indicates that it is not sufficient to address security alerts and vulnerabilities by forcing users to purchase upgrades.	3.1.1
C3	46% of diabetes-related LCS patients are unaware of whether their device can be updated or patched. This indicates that there is an awareness gap in LCS device documentation or training.	3.1.1
C4	72% of patients that do patch their diabetes-related LCS demonstrate caution or distrust of the patch. This indicates that there is a need for patch assurances.	3.1.1
C5	20% of diabetes-related LCS patients interact with or modify their devices through do it yourself (DIY) methods. 58% of those respondents report to be following the #wearenotwaiting effort. This indicates there is a significant patient demographic that relies on modifying their own LCS.	3.1.1

Finding #	Clinical Finding Text	Section
C6	There are multiple responses that indicate there is a broad cybersecurity assurance gap in diabetes-related LCS: • 90% of diabetes-related LCS patients report that there are no cybersecurity assurances for their device provided to them. • 65% of diabetes-related LCS doctors have concerns about the confidentiality, integrity, and availability of the devices they prescribe. • 59% of diabetes-related LCS patients have concerns about the confidentiality, integrity, and availability of data on their device(s). • 25% of diabetes-related LCS doctors report that their patients have also expressed concerns about the security of diabetes-related LCS. • 59% of diabetes-related LCS patients feel that even if a device is cleared by the FDA, it does not deliver sound cybersecurity.	3.1.1
C7	There are multiple responses that indicate there are patch validation gaps for diabetes-related LCS: • 36% of diabetes-related LCS patients report integrity and availability impacts caused by updating the mobile phone operating system software (e.g. Android/iOS) when they use it to support their continuous glucose monitor. • 31% of diabetes-related LCS patients report integrity and availability impacts caused by updating the continuous glucose monitor's mobile application. • 62% of diabetes-related LCS patients already do not always trust the data provided by their medical device, either because of availability or integrity issues. We believe this is primarily due to the existing inaccuracies in glucose measurement technologies and approaches.	3.1.1
C8	100% of diabetes-related LCS patients report strong physical security practices for their device use.	3.1.1

5.3 Industry Survey Findings

Finding #	Industry Finding Text	Section
I1	50% of the industry respondents are unaware of DIY efforts.	3.1.2
I2	50% of respondents report they have observed availability or integrity impacts with continuous glucose monitors from either phone operating system updates or mobile application updates. Some medical device manufacturers even track devices and equipment that are known to cause interference with their products[11].	3.1.2
I3	50% are concerned with the existing accuracy of their devices. This may be because the state of glucose measurement is an imperfect science.	3.1.2
I4	90% of respondents report they do think about the security of their devices.	3.1.2
I5	80% of respondents report they disable JTAG and other programming interfaces on the device once it is ready for the consumer.	3.1.2
I6	50% of respondents report they allow debugging of medical devices post-production.	3.1.2
I7	90% of respondents report that they intend for devices to be updated after they have been manufactured.	3.1.2

[11] http://www.medtronicdiabetes.com/customer-support/equipment-interference

5.4 Exploratory Research & Development Findings

Finding #	Exploratory R&D Finding Text	Section
E1	[redacted]	4.1
E2	[redacted]	4.1
E3	[redacted]	4.1
E4	[redacted]	4.1
E5	[redacted]	4.1
E6	We found that the currently embedded system virtualization available tools still require a large development investment for each available target while lacking fidelity and run-time efficiency. Specifically, Our research on the tools identified unwanted degradation in performance as compared to the physical device when research operated in hybrid emulation mode and no increase in performance when full emulation mode was compared to the physical device [168, 169].	4.2
E7	We identified the need for mature, feature-rich orchestration frameworks to support virtualization of LCS components and their I/O.	4.2
E8	We identified that the NRF51 DevKit is valid surrogate model development kit to use for further research relating to continuous glucose monitors.	4.2

Finding #	Exploratory R&D Finding Text	Section
E9	We determined that fuzzing virtualized LCS is feasible given a larger period of time dedicated to this goal.	4.2

5.5 Teardown Findings

Finding #	Teardown Finding Text	Section
T1	We identified architectures used in diabetes-related LCS environment, such as: • Vendor 1 has evolved primarily over the NRF 51 and NRF52 series of processors. • Components such as smartphones, tablets, and dedicated receivers commonly have other ARM processor variants. • Vendor 2 device series seems to use the msp430 processor.	4.3
T2	We encountered a steep level of hardware security measures which also prevent updates • The casing for the Vendor 1 device is designed so that the outer casing and the battery are close to the transmitter which has small wings. The wings secure the transmitter down inside the sensor. This setup makes it difficult to grind down the exterior shell. • The battery connector terminal is much smaller and thinner than previous Vendor 1 device versions. This changes makes it difficult to remove the battery without damaging its terminals. • Once the casing was removed from the transmitter, it was determined that the silicone dies were obfuscated with a stiff potting compound rendering identification incredibly challenging. • The Vendor 2 device series utilizes the msp430 microprocessor security bit to disable processor introspection via JTAG.	4.3
T3	The dedicated CGM receivers and transceivers are not intended to be updated or patched and do not have a working debug port such as JTAG.	4.3

Finding #	Teardown Finding Text	Section
T4	The associated smartphone applications are all updated through the respective device's application store.	4.3
T5	The pacemaker programmers do not utilize full disk encryption.	4.3
T6	The pacemakers themselves receive firmware updates from the programmers when necessary.	4.3
T7	The pacemaker programmers receive occasional updates from the manufacturer via signed usb drive.	4.3
T8	The pacemakers' PCBs often have internally exposed test points, which are available avenues for future proof of concept R&D efforts.	4.3
T9	The programmers generally leave functional debugging interfaces, which are available avenues for future proof of concept R&D efforts.	4.3
T10	The only processor commonalities we have identified so far in our LCS teardowns and in existing firmware analysis tools (Appendix A.1) is ARM. Future research for software and firmware analysis would benefit from identifying what existing analysis capabilities for ARM can be extended to LCS. However, we have not exhaustively analyzed a broad amount of LCS devices, just the most popular CGMs and pacemakers.	4.3

6 Synthesized Feasibility Study

This section presents our synthesized findings based on our literature review, clinical stakeholder findings, industry stakeholder findings, exploratory research findings, and teardown findings to synthesize an evidence based argument for future research and development avenues. Sections 6.1, 6.2, and 6.3 present recommendations based on the findings of our study. Finally we present other theoretical solutions and concepts that we have moderate confidence in which could be future work for exploratory research and development in Section 6.4.

In order to facilitate research and development per our recommendations, we advise the development of close relationships with medical device manufacturers. Due to the prevalence of disabling programming interfaces per finding I5, procuring special models of medical devices with JTAG and debugging interfaces left enabled or with additional introspection capabilities would greatly bolster 3rd party research and development for patch validation.

6.1 Controlling Risks: [redacted]

While testing software, firmware, and patches can help detect and eliminate vulnerabilities, there are opportunities for novel security controls to fundamentally limit the severity of such risks. Figure 4 illustrates two ways security controls can reduce overall risks in the FDA's Postmarket risk model [34].

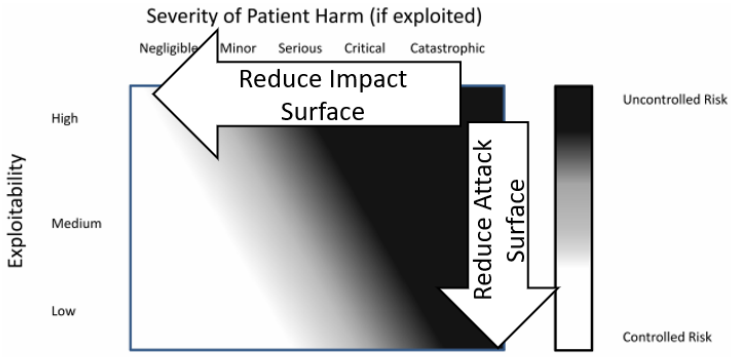

Figure 4: How novel security controls to reduce impacts and attacks can improve security with respect to the FDA Postmarket Management of Cybersecurity in Medical Devices risk assessment model. Model is from [34]

Existing security controls such as attack and exploit mitigations common on IT systems do not translate well to the embedded systems that make up LCS. [redacted].

6.1.1 [redacted] and Research

[redacted]

[redacted] Envisioned Clinical Use

[redacted]

[redacted]

- [redacted] (per finding E3).
- [redacted] (per finding L14).
- [redacted]
  - [redacted] E1).
  - [redacted].
- [redacted].
- [redacted] E5).
- [redacted] [17].

- [redacted].
- [redacted].

[redacted]. A motivating scenario is the KNOB attack [170], which currently is assumed to be pervasive across biomedical devices utilizing Bluetooth. [redacted].

6.1.2 [redacted] and Research

There currently are safemodes in implantable cardiac devices to support safe baseline heart pacing while the system is downloading a patch from a pacemaker programmer [7]. These safemodes simply keep pacing at 60 beats per minute and disable other capabilities. [redacted].

[redacted] Envisioned Clinical Use

[redacted].

- [redacted].
- [redacted].
- [redacted].

[redacted]

- [redacted]
- [redacted]

[redacted]

- "Implement device features that protect critical functionality and data, even when the device's cybersecurity has been compromised" from section V.B.3.a of [39].
- "The design should specify the level of autonomous functionality (resilience) any component of the system possesses when its communication capabilities with the rest of the system are disrupted including disruption of significant duration" from section V.B.3.c of [39].
- "Devices should be designed to be resilient to possible cybersecurity incident scenarios such as network outages, Denial of Service attacks, excessive bandwidth usage by other products, disrupted quality of service (QoS), and excessive jitter (i.e., a variation in the delay of received packets)." from section V.B.3.d of [39].

[redacted] Future Research

[redacted]

- [redacted]
- [redacted]
- [redacted]
- [redacted]

6.2 [redacted]

6.2 [redacted]

[redacted] (per finding C1), [redacted].

[redacted] [171].

[redacted].

[redacted]

[redacted] [171].

- [redacted].
- [redacted].
- [redacted].
- [redacted].

[redacted].

[redacted] Envisioned Clinical Use

[redacted]

- [redacted]
- [redacted]

[redacted] Future Research

[redacted]

- [redacted]
- [redacted]
- [redacted] L5.
- [redacted] [39].

[redacted]

[redacted]

- [redacted]
- [redacted]
- [redacted]

[redacted]

6.3 Software Testing at Scale: On High Fidelity Cyber Physical System Virtualization

6.3 Software Testing at Scale: On High Fidelity Cyber Physical System Virtualization

There are over 7000 medical device manufacturers in the FDA system, and to address the ever growing smart medical device market it is imperative that 3rd party software and patch validation testing tools be developed with scale, LCS composition diversity, and processor diversity in mind. Enterprise, desktop, and server software testing at scale in IT networks is typically achieved through regression, dynamic analysis, and dynamic symbolic analysis testing in virtualization environments. The existing tools for software and firmware testing of LCS are significantly limited in capabilities, performance, and fidelity, per finding E6. The diversity of processor and operating system architectures and I/O challenges in LCS are a significant barrier for applying such methodologies from the IT field to software and patch testing for LCS, per finding L14. However, there is emerging research that shows this barrier is surmountable.

Existing methodologies on how to achieve cyber physical system virtualization for select processor and instruction sets that specifically address the research barrier of the complexity of I/O to achieve virtualization and analysis capabilities which include: Redwood's SPAEVI methodology [172], which integrates high fidelity emulators and physics simulators; as well as EUROCOM S3's AVATAR and AVATAR2 methodologies [173, 107], which streams I/O from hardware-in-the-loop (HWIL) systems to cyber physical virtual machines over serial or JTAG. The AVATAR2 approach does not always achieve perfect streaming of I/O due to latency limitations over serial and JTAG. However, once captured I/O can be played in a loop for the target cyber physical system virtual machine for limited testing. Ultimately both published approaches face limitations stemming from using platforms and tools for emulation that are not purpose built to handle the nuanced I/O challenges of cyber physical systems or their components. Additionally, a hard challenge that remains unaddressed by literature is the virtualization and I/O coordination of sub-components, such as those common in LCS.

High Fidelity Cyber Physical System Virtualization Future Research

Furthering such advancements in SWIL/HWIL dynamic analysis and/or virtualization would significantly benefit 3rd party patch validation and software validation for LCS and similar fields. We believe this research is feasible, per finding E9, and future research should address:

- How the diversity of processor and operating system architectures for LCS will be addressed, per finding L14.
- How the challenges of I/O integration will be handled to simulate realistic physical behavior of the cyber-physical system, per finding L14, and at the multi-device scale, per findings L1 and E7.

6.4 Other Concepts for Future Work

6.4 Other Concepts for Future Work

This section contains recommendations that are backed by literature, clinical, industry, or teardown based findings, but lack concrete exploratory success.

6.4.1 New Machine-Readable Testing Resources and Models for LCS

We believe there is a case to make for research opportunities in researching and developing additional machine-readable documentation, models, testing, and system artifacts that might support 3rd party software and patch testing, based on:

- The FDA Premarket draft CBOM requirements [171].
- The large attack surface that make up multi-device LCS, per finding L4.
- The significant need for transparency of LCS hardware, firmware and software components, per finding L7.

Examples of resources that could be machine-readable include:

- Threat models (threat models are already in the draft FDA pre-market submission requirements);
- Regression tests;
- Network services and protocols (Bluetooth, Wi-Fi, IPv4/IPv6 + services);
- I/O descriptions (e.g. analog/digital signals, expected value ranges, etc.);
- I/O safety assumptions (e.g. Bio-Firewall rules);

The nature and format of such machine-readable resources would be a target for future research and development, and has parallels in other research programs [174].

6.4.2 Secure Containerization of LCS Software

Many vendors are looking at containerization to speed up software development and testing, while achieving greater security. However, recent analysis has shown that the majority, over 60%, of the top available docker containers contain significant vulnerabilities[175]. Containerization of LCS software would simplify patch validation needs, as individual modules are generally less complex to test than changes to a monolithic system or firmware blob. We believe the most likely platforms to support software containerization are infusion pumps, which often are made of many discrete sub-components that talk over serial, CANbus, or USB busses. Additionally, there may be software and patch risk control research and development opportunities per container, such as Bio-Firewalls per containers.

6.4.3 P-Code Based Crude Cyber Physical System Virtualization

The National Security Agency (NSA) recently publsished a static analysis tool named Ghidra which offers a universal intermediate representation model called "p-code" which allows for decompilation of binaries. It has been announced that Ghidra will later support dynamic analysis capabilities such as debugging in a future release. This may be a viable alternative to high fidelity cyber physical system virtualization for limited dynamic analysis, and there exists skeleton code to suggest it may already be a consideration for the Ghidra project, which we discuss below.

We believe the p-code could be a language to support universal emulation of software and some firmware across Levels 4,5,6 (e.g. Object-oriented programming language, problem-oriented language, and Assembly language) of the Bhunia et. al. model for six levels of software on top of digital logic hardware [86]. It would not be able to support virtualization of device I/O and I/O-related code due to its limitations. P-code does not cover machine specific instructions or registers, which prevents p-code from offering static or dynamic analysis capabilities for code that support device I/O. This approach would require a p-code emulation engine to be developed. There is an undocumented p-code emulator in the Ghidra public code repository. The current emulator code[12] is currently not driven by any Ghidra code and p-code bindings would have to be written to drive it per instruction.

It should be noted that this approach specifically will not address the complexity of I/O, and would require a high degree of informal methods to achieve success.

6.4.4 Modular Based LCS Design

Modular based system design, in an ideal setting, allows for the replacement of precisely-understood modules that have no side-effects in function or interaction. This research avenue presents a wide variety of opportunities such as high-assurance heterogeneous consensus-based systems, similar to Apollo lander computer; plug-and-play components for LCS; and modular prosthetics.

[12] `https://github.com/NationalSecurityAgency/ghidra/blob/master/Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/pcode/emulate/Emulate.java`

Modular based LCS design research and development currently would be better suited for life-critical infrastructure systems, such as computerized tomography (CT) scanners; Magnetic resonance imaging (MRI) systems; as well as robotic surgical equipment and other fixed, imaging, diagnostic, or treatment infrastructure; however, future medical devices such as modular prosthetic limbs[176] and brain-to-machine neural-links[177] could also be ideal for modular based design to offer greater safeties and assurances.

The benefits of this approach would allow for simpler component-based patch validation, but may increase the complexity of system-of-system level patch validation efforts if component interaction aspects are patched. Beyond patch validation, modular design in general may allow for greater cybersecurity attack survivability to be achieved for critical systems over monolithic designs.

7 Patch Assurance Complexity Model

Finding L13 suggests that the state of the art for type 2-4 patch validation of LCS is immature, as described in Section 1.2. Mature solutions, already exist for area 1, such as hash checking tools, medical device patch hash validation services [178], and software signatures. Both literature survey and clinical survey present the need for high-assurance patch validation capabilities to address the gaps for LCS, while our exploratory research offer possible technologies that could reduce patch validation difficulty and complexity.

We present a model below for high assurance patch validation for devices based on multiple degrees of trustworthiness for patch validation efforts. The factors that drive the complexity of performing high-assurance patch validation are numerous. Our model's objective is to capture and enable navigation of the primary common challenges in patch validation to inform future research and development.

7.1 High Assurance Patch Validation Complexity Model

Our research identified numerous risks in navigating the technical challenges of researching and developing new techniques to support high assurance patch validation. But it seems feasible with the right framework to manage these risks. To support this high-risk, high-reward research area, we have developed a holistic high assurance patch validation complexity model.

The model quantitatively delineates the complexity of this problem with respect of the complexity to validate an arbitrary patch for a given device. It then offers scoring metrics to allow for quantitative scoring of high assurance patch validation complexity. We believe this approach will allow for the tractable, productive research and development of solutions for this generally intractable problem. This model allows you to judge the complexity of research and development automation of high assurance patch validation for a device across hardware patches, firmware patches, and software patches. This model is composed of the following high-level dimensions:

- **Reprogrammability** - Reprogrammability directly affects the assurances we can have with validating a patch. Patching the system in a manner that the device was designed to support allows for more assurances than informal techniques or unintended techniques to patch the

system. Many LCS that predate the current era of FDA requirements are only partially reprogrammable or not at all, which will directly affect patch development and patch validation. This dimension is derived from findings L1, I7, T3, T4, T5, T8 and T9.

- **Patch risk controls** – The degree to which safety controls have been engineered to limit dangerous operation of the device may also limit risks to dangerous operations caused by a patch. Such controls that limit patching risks offer assurances for patch-validation efforts. This dimension is derived from finding L2, L9, E2 and E5.
- **Introspection capabilities** – The degree to which an end user can observe changes to the device's functionality directly influences the assurances we can have in patch validation efforts. If a patch tester or end user cannot observe meaningful changes to a blackbox system when patched (e.g. a simple indicator showing a successful patch), there are fundamental limits on the trust they should have on the patching of said system. This is one of the driving reasons why clinics commonly decline to patch implantable LCS for patients to date. The greater degree of transparency for a system, usually lends to a greater degree of assurances there are in patching the system. Introspection capabilities includes access to device logs, the device's ability to be virtualized, availability of phone or cloud apps that integrate with the device, etc. This is derived from findings L10, L11, I5, T6, T11 and T12.
- **Available development artifacts available for 3rd party testing** - Available artifacts for the patch validation 3rd party to utilize for testing such as regression tests offer assurances to the patch validation effort. Regression tests and use-case tests are essential for validating intended use of the device. This dimension is derived from findings L4, L6, L7, L8, L11 and L12.
- **Security controls noninterference for patch testing** - Security controls can often impede or hinder patch testing efforts. While some security controls can be disabled to support patch testing efforts, some cannot and may interfere. To accurately measure the effort required for patch validation assurance, we must capture and rank these security controls. This dimension is derived from findings I5 and our experiences in testing embedded system software.

This model leverages recent work presented in [102]. We present a quantitatively-scored Excel worksheet as an attachment to this study. It is initialized with a toy example to simply illustrate the scoring in Figure 5. To address finding L1, this model would be completed for each major component in a LCS.

The following subsections explain how the model measures patch assurance and validation complexity for software patches, firmware patches, and hardware patches to LCS.

7.2 Software Patch Assurance and Validation Complexity

Software patches include but are not limited to: modification of device application(s), modification of 3rd party libraries, modification of extra-device supporting software (e.g. phone or cloud applications), etc. A radar chart illustration of these factors in the model is presented in Figure 6.

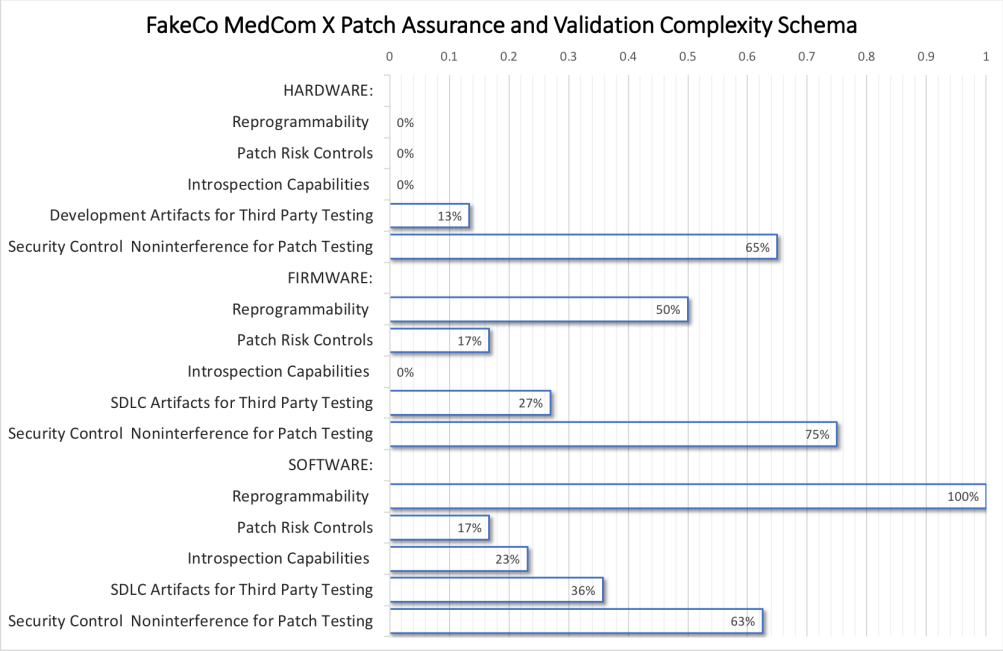

Figure 5: High Assurance Patch Validation Complexity Model. Fictional device "FakeCo MedCom X" is for illustration purposes only.

7.2.1 Software Reprogrammability

At this level the model considers how reprogrammable the software stack is for the device.

7.2.2 Software Patch Risk Controls

At this level the model considers what software-based safety mechanisms exist that could mitigate or limit the risks a future patch could pose. Mechanisms, tools, techniques, and practices here include: [redacted] (e.g. [redacted] Section 6.1.1); [redacted]; application trust levels; use of least-privilege principle in application design; and tools, techniques and practices for ensuring critical task scheduling (e.g. watchdog timers, critical task schedulers, etc.).

7.2.3 Software Introspection Capabilities

At this level the model considers how observable or transparent changes to the system are to various user groups. Software-introspection capabilities here include for the following end-users:

- Patients
  - Audio/visual indicators on device for software factors

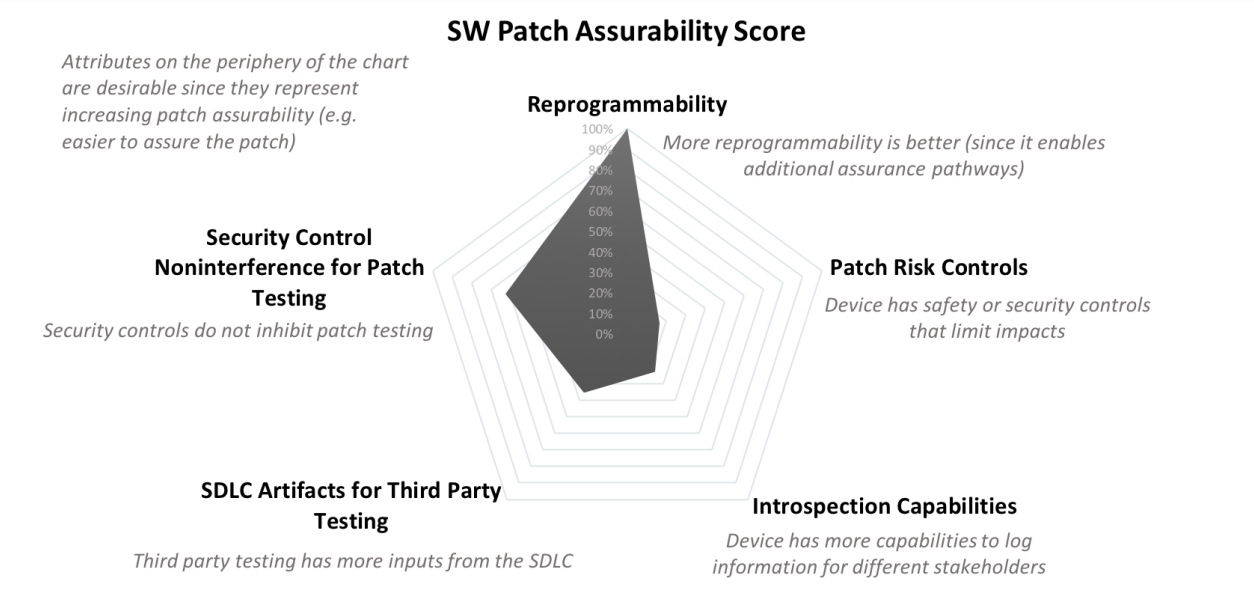

Figure 6: The Software Patch Assurability Score from the High Assurance Patch Validation Complexity Model. Fictional device "FakeCo MedCom X" is for illustration purposes only.

  - Mobile application or web service accessible device software logs

- Care Providers / Emergency Medical Technicians
  - In-situ device operational software logs and analytics
  - In-situ device safemode operational software logs

- Researchers / Forensic Analysts
  - Cyber range integrability for high fidelity operational and performance SWIL (software in the loop) testing
  - I/O fuzzing capabilities for system
  - Cyber physical system virtualizability

7.2.4 Development Life Cycle Artifacts for 3rd Party Testing

At this level the model considers the availability of development testing resources for 3rd party patch validation efforts. Resources are not scored if they are unavailable to the 3rd party performing the patch validation efforts. For the device's software these resources include:

- Whitebox testing resources:

  - Open source or available source code
  - Use of common criteria design artifacts
  - Regression testing artifacts
  - Design-to-code model-based formal verification artifacts
  - Source-based static analysis testing artifacts
- Graybox testing resources:
  - Cybersecurity Bill of Materials (CBOM)
  - Software Bill of Materials (SBOM)
  - FDA-compliant threat models
  - Binary-based static analysis testing artifacts
  - Binary-based symbolic/concolic testing artifacts
  - Dynamic analysis testing artifacts

It should be noted that even in the presence of these resources, their accuracy, details and/or usability may be limited. The FDA in October 2018 introduced the concept of a machine readable, cybersecurity bill of materials (CBOM) for medical devices, which would be "a list of commercial, open source, and off-the-shelf software and hardware components to enable device users — including patients, care providers, and healthcare delivery organizations (HDOs) — to effectively manage their assets, understand the potential impact of identified vulnerabilities to the device – and the connected system – and to deploy countermeasures to maintain the device's essential performance" [39].

However, there are practical limitations to the details and usability of the two current SBOM standards which the CBOM may utilize, as they do not cover component/sub-components of systems such as software libraries, and firmware libraries. Additionally due to intellectual property licenses, restrictions, and agreements it may not be possible for manufacturers to provide a fully completely CBOM anytime soon. Sub-component and library information is pertinent in determining the applicability of software library vulnerability disclosures to medical devices, without which diminishes the intended goal of CBOM [41].

7.2.5 Software Security Controls Noninterference for Patch Testing

At this level the model considers which software-based security controls can impede or hinder patch testing efforts. The model considers controls if they are present AND they interfere with patch validation efforts. If security controls can be disabled to not interfere with patch validation efforts, they are not scored in this model. These include but are not limited to: the use of strong authentication for system services and protocols; use of cryptography for services and protocols; use of ingress and egress firewalls for system communication; use of application signatures; use of memory execution protections (e.g. Data Execution Prevention (DEP), "Not Execute"(NX), "Write Not Execute"); use of control flow integrity tools, techniques, and practices (e.g. retpoline,

RELRO, shadow stack, stack cookies); and use of obfuscation-based exploit mitigation techniques (e.g. Address Space Layout Randomization (ASLR), pointer encoding, pointer encryption, etc.).

The model tallies up interfering security controls and inversely represents it as the degree of non-interference to patch validation efforts.

7.3 Firmware Patch Assurance and Validation Complexity

Firmware patches include but are not limited to driver modifications, BIOS/UEFI firmware changes, and secondary chip firmware modifications (e.g. Bluetooth communication microprocessor flashing). A radar chart illustration of these factors in the model is presented in Figure 7.

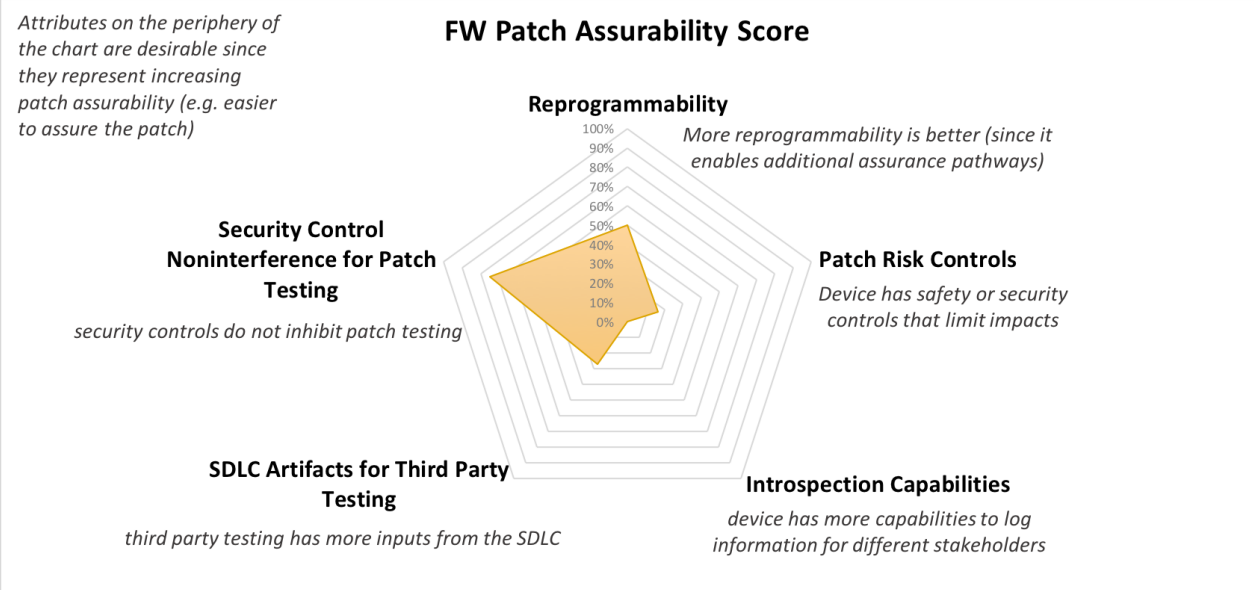

Figure 7: The Firmware Patch Assurability Score from the High Assurance Patch Validation Complexity Model. Fictional device "FakeCo MedCom X" is for illustration purposes only.

7.3.1 Reprogrammability

At this level the model considers how reprogrammable the firmware stack is for the device.

7.3.2 Patch Risk Controls

At this level the model considers what firmware-based safety mechanisms exist that could mitigate or limit the risks a future patch at this level, or lower levels could pose. Mechanisms and TTPs here include: [redacted] (e.g. [redacted] Section 6.1.1); [redacted]; application trust levels; use of least-privilege principle in

application design; and TTPs for ensuring critical task scheduling (e.g. watchdog timers, critical task schedulers, etc.).

7.3.3 Introspection Capabilities

At this level the model considers how observable or transparent changes to the system are to various user groups. Firmware-introspection capabilities here include for the following end-users:

- Patients
  - Audio/visual indicators on device for firmware factors
  - Mobile application or web service accessible device firmware logs
- Care Providers / Emergency Medical Technicians
  - In-situ device operational firmware logs and analytics
  - In-situ device safemode operational firmware logs
- Researchers / Forensic Analysts
  - Cyber range integrability for high fidelity operational and performance FWIL (firmware in the loop) testing
  - I/O fuzzing capabilities for system
  - Cyber physical system virtualizability

7.3.4 Development Life Cycle Artifacts for 3rd Party Testing

At this level the model considers the availability of development testing resources for 3rd party patch validation efforts. Resources are not scored if they are unavailable to the 3rd party performing the patch validation efforts. For the device's firmware these resources include:

- Whitebox testing resources:
  - Open source or available source code
  - Use of common criteria design artifacts
  - Regression testing artifacts
  - Design-to-code model-based formal verification artifacts
  - Source-based static analysis testing artifacts
- Graybox testing resources:
  - Software Bill of Materials (SBOM) covering firmware and drivers
  - FDA-compliant threat models

- Binary-based static analysis testing artifacts
- Binary-based symbolic/concolic testing artifacts
- Dynamic analysis testing artifacts

7.3.5 Security Controls Noninterference for Patch Testing

At this level the model considers which firmware-based security controls can impede or hinder patch testing efforts. The model considers controls if they are present AND they interfere with patch validation efforts. If security controls can be disabled to not interfere with patch validation efforts, they are not scored in this model. These include but are not limited to: use of authentication, permissions and/or hardening of Basic Input Output System (BIOS) or Universal Extensible Firmware Interface (UEFI); use of cryptography in radiofrequency communication with system peripherals and components; use of tamper-proof-mechanisms (TPM) in peripherals; use of secure bootloader and firmware signatures; use of driver signatures; use of control flow integrity TTPs (e.g. shadow stack, memory segmentation, etc.); use of memory execution protections (e.g. Data Execution Prevention (DEP), "Not Execute"(NX), "Write Not Execute"($W\hat{X}$)); and use of obfuscation-based exploit mitigation techniques (e.g. Address Space Layout Randomization (ASLR), pointer encoding, pointer encryption, etc.).

The model tallies up interfering security controls, and inversely represents it as the degree of non-interference to patch validation efforts.

7.4 Hardware Patch Assurance and Validation Complexity

Hardware patches include but are not limited to jumper reconfiguration, module installation, component replacement, and FPGA reprogramming. Fixed form systems are typically less reprogrammable at the hardware level than modular systems. Hardware patch validation is the hardest category of patch validation, and some patch types may be impossible or infeasible to validate, for instance processor microcode patch validation is likely beyond the capabilities of a 3rd party to validate. A radar chart illustration of these factors in the model is presented in Figure 8.

7.4.1 Reprogrammability

Hardware Reprogrammability directly affects the assurances we can have with validating a hardware-based patch. Patching the hardware in a manner that the device was designed to support allows for more assurances that informal techniques or unintended techniques to patch the hardware.

7.4.2 Patch Risk Controls

At this level the model considers what hardware-based safety mechanisms exist that could mitigate or limit the risks a future patch at this level, or lower levels could pose. Mechanisms, tools, techniques, and practices here include: filtering of exposed I/O for anomalies (e.g. Bio-Firewall concept in Section 6.1.1), hardware based safemodes, and secure storage for trusted applications.

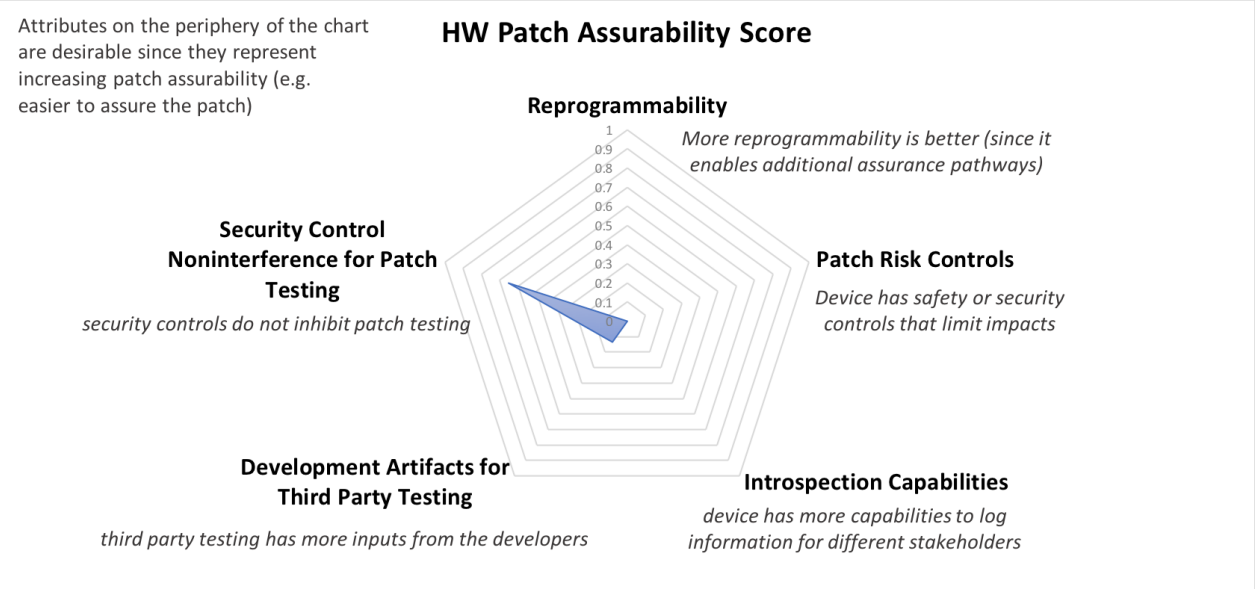

Figure 8: The Hardware Patch Assurability Score from the High Assurance Patch Validation Complexity Model. Fictional device "FakeCo MedCom X" is for illustration purposes only.

7.4.3 Introspection Capabilities

At this level the model considers how observable or transparent changes to the system are to various user groups. Hardware-introspection capabilities here include for the following end-users:

- Patients
  - Audio/visual indicators on device for hardware factors
  - Mobile application or web service accessible device hardware logs
- Care Providers / Emergency Medical Technicians
  - In-situ device operational hardware logs and analytics
  - In-situ device safemode operational hardware logs
- Researchers / Forensic Analysts
  - Cyber range integrability for high fidelity operational and performance HWIL (hardware in the loop) testing
  - I/O fuzzing capabilities for system
  - Cyber physical system virtualizability

7.4.4 Development Life Cycle Artifacts for 3rd Party Testing

At this level the model considers the availability of development testing resources for 3rd party patch validation efforts. Resources are not scored if they are unavailable to the 3rd party performing the patch validation efforts. For the device's hardware these resources include:

- Whitebox testing resources:
  - Open source or available source hardware design
  - Regression testing artifacts
  - Design-to-code model-based formal verification artifacts
- Graybox testing resources:
  - Hardware bill of materials (HBOM)
  - FDA-compliant threat models
  - Device certification testing artifacts

7.4.5 Security Controls Noninterference for Patch Testing

At this level the model considers which hardware-based security controls can impede or hinder patch testing efforts. The model considers controls if they are present AND they interfere with patch validation efforts. If security controls can be disabled to not interfere with patch validation efforts, they are not scored in this model. These include but are not limited to: use of cryptography on device bus; use of TPM on peripherals; use of Joint Test Action Group (JTAG) authentication for hardware access control; disabling JTAG to prevent hardware access; use of Serial Communications Interface (SCI) authentication for hardware access control; disabling SCI to prevent hardware access; use of microprocessor security bits for hardware access control; number of hardware supported control flow integrity TTPs (e.g. shadow stack, memory segmentation, etc.); analog based security TTPs (e.g. DARPA Leveraging the Analog Domain for Security (LADS) TTPs); and use of anti-tamper seals and device mechanisms. Note that disabling hardware access incurs a penalty for device introspection.

The model tallies up interfering security controls, and inversely represents it as the degree of non-interference to patch validation efforts.

7.5 Utilizing this Model to Analyze Assurance and Validation Complexity for a Patch

This model allows one to analyze what challenges and supporting resources exist for patch validation across software, firmware and hardware. At a high level, the quantitative scoring scheme gives a preliminary indication on the complexity the patch validation task will involve, and supports the development of qualitative thresholds to avoid attempting overly complex patch validation, which may be key for automation.

Finally, while no two patches are alike, the model's individual worksheets (see Study attachment HAPVCSv1.7.xlsx) allow the patch tester to identify specific sources of challenges or supporting resources or capabilities for patch validation, which can guide and streamline initial patch validation planning.

Beyond that, the patch validation efforts should be targeted at the contents of the patch itself, as well as the aspects of the specified functionality of the target code or hardware to be patched, and is outside the scope of this model. Fruitful future work with the model may include developing patch validation processes, playbook generation capabilities, guidance generation capabilities, and artificial intelligence (e.g. automated patch generation and validation) capabilities.

8 A Future for Novel Clinical Cybersecurity Capabilities

[redacted]
Section 6, [redacted]. [redacted]. [redacted].

Figure 9: [redacted]

9 Concluding Remarks

In conclusion, there are many challenges to improving the cybersecurity hygiene of biomedical devices, and it is not as simple as instructing everyone to patch their devices as often as they wash their hands. We have presented a holistic review of these challenges and offer multiple recommendations for advancing the both hygiene and science of securing life-critical systems.

We have presented literature, clinical, and exploratory applied research findings to make evidence based recommendations on feasible future research and development to advance the science of securing life critical systems. Specifically we presented several avenues of research and development supported by combined literature, stakeholder, teardown, and/or exploratory findings.

We presented a model for scoring the high assurance patch validation complexity for life critical systems, which may be helpful in guiding future patch validation research and development efforts. This is a flexible findings-based and component-centric model to quantitatively score the complexity of general patch validation for LCS, to advise future patch validation research and development on the risks for any given LCS platform.

These combined findings, contributions, and recommendations will advance the science of patching life critical systems by reducing risks, increasing patch validation capabilities, increasing attack recovery and resilience capabilities, and most importantly, building up the lacking trust in the clinical stakeholder community for patching life-critical systems. Finally we demonstrated a future for securing life-critical biomedical devices, that presents novel clinical capabilities for biomedical device cyberattack detection, prevention, and response that can be achieved after biomedical cybersecurity hygiene is improved.

A Existing Firmware Analysis Tools

This subsection attempts to cover a survey of available dynamic, symbolic/concolic, and static analysis tools primarily found in works analyzing firmware.

A.1 Firmware Analysis Tools

Below is a brief survey of popular firmware analysis tools. It should be noted that any firmware analysis tool utilizing QEMU (Quick EMUlator) generally boasts support of ARM, ARM64, M68k, MIPS, SPARC, and x86 emulation, but may only offer analysis for a subset of architectures.

- **AFL** ("American fuzzy lop") is a security-oriented fuzzer that employs compile-time instrumentation and genetic algorithms to automatically discover clean, test cases that trigger new internal states in the targeted binary to maximize code coverage in testing [105]. It offers experimental support for blackbox fuzzing of binaries, without compiler hooks by means of emulation hooks, through QEMU. It has been integrated with many other platforms, such as Unicorn (AFL-Unicorn).

- **Avatar2** is a python2-based orchestration framework which is often used in conjunction with AFL-Unicorn, PANDA, QEMU, and a debugger with physical hardware to produce HWIL testing environments which can record and repeat segments of execution in order to perform an effective depth-oriented coverage-based analysis [168, 173]. While this sounds ideal, a substantial amount of effort is still involved in establishing a fuzzing environment in Avatar2 which, coupled with the performance impacts involved in the use of Python and the approaching end of life for Python 2, makes this tool less attractive. There are also regularly reported performance degradations caused by the hybrid emulation HWIL configurations.

- **Firmadyne** is a collection of tools for linux-based embedded firmware analysis at many levels [179] and is based on QEMU. By integrating directly into the linux kernel, FIRMADYNE achieves an impressive level of operating system-wide coverage at the cost of runtime efficiency from the bloated kernel with unused drivers and extra instrumentation. While many embedded systems are linux-based, this tool is not applicable on many LCS targets when a linux kernel is not available. Many of the preprocessing tools such as the scraper which downloads firmware from manufacturer websites, firmware extractor which extracts kernels and filesystems, and the NVRAM emulation C library may come in use for other projects in the future.

- **Firmware Slap** combines concolic analysis with function clustering for vulnerability discovery and function similarity in firmware. It is built as a series of libraries and exports most information as either Python pickles or JSON for integration with other tools [180, 181]. It boasts a simple python interface, is developed for Python 3.6, and is integrated with Ghidra and RADARE2[13]. It was recently released at DEFCON 27, and it would be interesting to

[13] https://www.radare.org/r/

analyze the output of this tool against that of pure dynamic analysis tools with correlations across the spectra of call-depth, vulnerability type, and application segment. The overall workqueue structure of this project can also be an efficient way to organize a myriad of asynchronous tasks in this problem space.

- **PANDA** is a dynamic analysis platform based on QEMU which allows the ability to record and replay executions in a compact sharable way to facilitate repeatable experiments while maintaining a consistent interface across all target architectures of QEMU [182]. Moreover, this project boasts the performance benefits of C while maintaining wide integration with Avatar2. This allows for HWIL testing and PANDA is a simple choice for further emulation efforts related to architectures emulated by QEMU. Reliance upon QEMU leaves the product suboptimal for LCS which so often run on esoteric architectures with poor support from QEMU.
- **Unicorn Engine** is designed to be a lightweight multi-platform, multi-architecture CPU emulator framework, that is based on QEMU [183], is implemented in pure C, and offers bindings for various scripting languages. These bindings have been leveraged by many software analysis tools such as the AFL-Unicorn project.

A.2 Relevant Utilities

- **Deepstate** is a "framework that provides C and C++ developers with a common interface to various symbolic execution and fuzzing engines" [184]. DeepState is an orchestration tool for common fuzzers with a gtest-like interface from C and C++. Supporting a wide array of test generation backends (Manticore, Angr, LibFuzzer, AFL, Eclipse, etc) there are plenty of examples to use when generating new backends for the tool. With API sequence test support, automated test generation, and the performance benefits of C and C++, it offers promising capabilities for LCS.
- **Binwalk** is a static analysis tool for recognizing artifacts of compression, filesystems, files, and libraries in firmware [185]. It does not provide code analysis capabilities, but is used for providing information and metadata on firmware.

A.3 Relevant Non-Firmware Tools

This section contains noteworthy static, dynamic, and/or symbolic analysis approaches in the x86 / ARM space that can offer lessons for software testing in firmware analysis tool research and development.

- **Angr** is a Python framework for analyzing binaries that combines both static and dynamic symbolic ("concolic") analysis, making it applicable to a variety of tasks [186]. It has been maintained and upgraded to support Python 3. Angr was a key part of the DARPA Cyber Grand Challenge 3rd place platform "Mechanical Phish" and offers the following capabilities:
  - Control-flow graph recovery.
  - Symbolic execution.
  - Automatic ROP chain building.
  - Automatic binary hardening and patch generation.
  - Automatic exploit generation (for CGC DECREE and simple linux binaries).

  Angr's primary weaknesses for firmware analysis are that common libraries in analysis are replaced by buggy or incomplete "SimProcedures", and it does not support kernel system calls[14].

- **GRR** is a powerful fuzzer and emulator for x86 and AMD64 targets through similar record/replay depth traversal as Avatar2 with Panda. Created to fuzz binaries for the Cyber Grand Challenge [187], the focus of this project is on general purpose machines making it difficult to use for embedded reverse engineering. GRR is an x86 to amd64 binary translator designed to emulate and fuzz DECREE challenge binaries automatically.

- **S2E** is a platform that allows for the writing of tools that can analyze the properties and behaviors of software systems [188]. It also was used in the Cyber Grand Challenge[15] and it has the ability to run unmodified x86, x86-64, or ARM software stacks, including programs, libraries, the kernel, and drivers. S2E's strengths include ability to quickly find vulnerabilities in software stacks, a maintained codebase, designed for ease-of-use, and emulates the KVM interface and is easily adaptable to any KVM-based virtualization. S2E's weaknesses include that it does not support embedded instruction sets or environments by default.

- **libfuzzer** is an "in-process, coverage-guided, evolutionary fuzzing engine" built by the LLVM compiler project [189]. It acts as a test suite built into the binary that works on a per-function level. It is a tool for white-box or gray-box testing of software.

- **PySymEmu** is a symbolic execution tool, capable of automatically generating interesting inputs for x86/x64 binary programs [190]. Its weaknesses are that it is in alpha status, only supports linux binaries in x86/64 and the last update was over 4 years ago.

[14] `https://docs.angr.io/advanced-topics/gotchas`
[15] `http://s2e.systems/docs/Tutorials/PoV/index.html`

B References

[1] FDA, "Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices," 2005-05-11. [Online]. Available: https://www.fda.gov/regulatory-information/search-fda-guidance-documents/guidance-content-premarket-submissions-software-contained-medical-devices

[2] M. Kintzlinger and N. Nissim, "Keep an eye on your personal belongings! The security of personal medical devices and their ecosystems," *Journal of Biomedical Informatics*, vol. 95, p. 103233, Jul. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1532046419301522

[3] P. A. Williams and A. J. Woodward, "Cybersecurity vulnerabilities in medical devices: a complex environment and multifaceted problem," *Medical Devices (Auckland, N.Z.)*, vol. 8, pp. 305–316, 2015. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4516335/

[4] A. Baranchuk, M. M. Refaat, K. K. Patton, M. K. Chung, K. Krishnan, V. Kutyifa, G. Upadhyay, J. D. Fisher, and D. R. Lakkireddy, "Cybersecurity for Cardiac Implantable Electronic Devices: What Should You Know?" *Journal of the American College of Cardiology*, vol. 71, no. 11, pp. 1284–1288, Mar. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0735109718302006

[5] C. Kersbergen, *Cardiac Defibrillators Need to Have a Bulletproof Vest: The National Security Risk Posed the Lack of Cybersecurity in Implantable Medical Devices*, 2016. [Online]. Available: https://heinonline.org/HOL/P?h=hein.journals/novalr41&i=437

[6] J. A. Hansen and N. M. Hansen, "A taxonomy of vulnerabilities in implantable medical devices," in *Proceedings of the second annual workshop on Security and privacy in medical and home-care systems*. ACM, 2010, pp. 13–20. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/1866914.1866917

[7] M. Mirowski, M. M. Mower, W. S. Staewen, B. Tabatznik, and A. I. Mendeloff, "Standby automatic defibrillator: an approach to prevention of sudden coronary death," *Archives of Internal Medicine*, vol. 126, no. 1, pp. 158–161, 1970. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/5425512

[8] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, "Cortical control of a prosthetic arm for self-feeding," *Nature*, vol. 453, no. 7198, p. 1098, 2008. [Online]. Available: https://academic.oup.com/neurosurgery/article/63/2/N8/2558082

[9] R. B. North, D. H. Kidd, M. Zahurak, C. S. James, and D. M. Long, "Spinal cord stimulation for chronic, intractable pain: experience over two decades," *Neurosurgery*, vol. 32, no. 3, pp. 384–395, 1993. [Online]. Available: https://academic.oup.com/neurosurgery/article-abstract/32/3/384/2755019

[10] D. Song, R. H. Chan, V. Z. Marmarelis, R. E. Hampson, S. A. Deadwyler, and T. W. Berger, "Nonlinear modeling of neural population dynamics for hippocampal prostheses," *Neural Networks*, vol. 22, no. 9, pp. 1340–1351, 2009. [Online]. Available: https://ieeexplore.ieee.org/document/6171069

[11] A. Y. Chow, V. Y. Chow, K. H. Packo, J. S. Pollack, G. A. Peyman, and R. Schuchard, "The artificial silicon retina microchip for the treatment of visionloss from retinitis pigmentosa," *Archives of Ophthalmology*, vol. 122, no. 4, pp. 460–469, 2004. [Online]. Available: https://jamanetwork.com/journals/jamaophthalmology/fullarticle/416206

[12] J. Halamka, A. Juels, A. Stubblefield, and J. Westhues, "The security implications of verichip cloning," *Journal of the American Medical Informatics Association*, vol. 13, no. 6, pp. 601–607, 2006. [Online]. Available: https://academic.oup.com/jamia/article/13/6/601/735450

[13] B. Bitarello, H. Fuks, and J. Queiroz, "New technologies for dynamic tattoo art," in *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*. ACM, 2011, pp. 313–316. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/1935701.1935774?download=true

[14] D. Raskovic, T. Martin, and E. Jovanov, "Medical monitoring applications for wearable computing," *The Computer Journal*, vol. 47, no. 4, pp. 495–504, 2004. [Online]. Available: http://www.ece.uah.edu/~jovanov/papers/J2004_Raskovic_Wearable.pdf

[15] M. Alrige and S. Chatterjee, "Toward a taxonomy of wearable technologies in healthcare," in *International Conference on Design Science Research in Information Systems*, 2015, pp. 496–504.

[16] R. T. Azevedo, N. Bennett, A. Bilicki, J. Hooper, F. Markopoulou, and M. Tsakiris, "The calming effect of a new wearable device during the anticipation of public speech," *Scientific reports*, vol. 7, no. 1, p. 2285, 2017. [Online]. Available: https://www.researchgate.net/publication/317192192_The_calming_effect_of_a_new_wearable_device_during_the_anticipation_of_public_speech

[17] G. Rebel, F. J. Estevez, and P. Gloesekoetter, "Energy efficiency study of representative microcontrollers for wearable electronics," in *International Conference on Bioinformatics and Biomedical Engineering*. Springer, 2015, pp. 65–76. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-16480-9_7

[18] B. W. Bequette, "A critical assessment of algorithms and challenges in the development of a closed-loop artificial pancreas," *Diabetes technology & therapeutics*, vol. 7, no. 1, pp. 28–47, 2005. [Online]. Available: https://pdfs.semanticscholar.org/2407/55fdb9a1b11f962b6558ba307f5d660b7a83.pdf

[19] J. A. Pineda, "Method and system for predicting and preventing seizures," Sep. 11 2007, uS Patent 7,269,455. [Online]. Available: https://patents.google.com/patent/US7269455B2/en

[20] V. K. Varadan, "Wearable remote electrophysiological monitoring system," Oct. 24 2013, uS Patent App. 13/449,755. [Online]. Available: https://patents.google.com/patent/US20130281815

[21] C. P. Groff and P. L. Mulvaney, "Wearable vital sign monitoring system," Aug. 15 2000, uS Patent 6,102,856. [Online]. Available: https://patentimages.storage.googleapis.com/63/4f/01/c89db91ac07563/US6102856.pdf

[22] G. Li, B.-L. Lee, and W.-Y. Chung, "Smartwatch-based wearable eeg system for driver drowsiness detection," *IEEE Sensors Journal*, vol. 15, no. 12, pp. 7169–7180, 2015. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7236886

[23] C.-T. Lin, C.-H. Chuang, C.-S. Huang, S.-F. Tsai, S.-W. Lu, Y.-H. Chen, and L.-W. Ko, "Wireless and wearable eeg system for evaluating driver vigilance," *IEEE Transactions on biomedical circuits and systems*, vol. 8, no. 2, pp. 165–176, 2014. [Online]. Available: https://ir.nctu.edu.tw/bitstream/11536/24756/1/000337154000003.pdf

[24] F. Rohit, V. Kulathumani, R. Kavi, I. Elwarfalli, V. Kecojevic, and A. Nimbarte, "Real-time drowsiness detection using wearable, lightweight brain sensing headbands," *IET Intelligent Transport Systems*, vol. 11, no. 5, pp. 255–263, 2017. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7927523

[25] S. Patel, R. Hughes, T. Hester, J. Stein, M. Akay, J. G. Dy, and P. Bonato, "A novel approach to monitor rehabilitation outcomes in stroke survivors using wearable technology," *Proceedings of the IEEE*, vol. 98, no. 3, pp. 450–461, 2010. [Online]. Available: https://dspace.mit.edu/bitstream/handle/1721.1/62167/Patel-2010-A%20Novel%20Approach%20to%20Monitor%20Rehabilitation%20Outcomes%20in%20Stroke%20Survivors%20Using%20Wearable%20Technology.pdf?sequence=2&isAllowed=y

[26] G. Valenza, M. Nardelli, A. Lanata, C. Gentili, G. Bertschy, R. Paradiso, and E. P. Scilingo, "Wearable monitoring for mood recognition in bipolar disorder based on history-dependent long-term heart rate variability analysis," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 5, pp. 1625–1635, 2013. [Online]. Available: https://ieeexplore.ieee.org/document/6661378

[27] M. Rofouei, M. Sinclair, R. Bittner, T. Blank, N. Saw, G. DeJean, and J. Heffron, "A non-invasive wearable neck-cuff system for real-time sleep monitoring," in *2011 international conference on body sensor networks*. IEEE, 2011, pp. 156–161. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5955315

[28] Center for Devices and Radiological Health, "Overview of medical device classification and reclassification," 2017. [Online]. Available: https://www.fda.gov/about-fda/cdrh-transparency/overview-medical-device-classification-and-reclassification

[29] B. medical, "What's the Difference Between the FDA Medical Device Classes?" [Online]. Available: https://www.bmpmedical.com/blog/whats-difference-fda-medical-device-classes-2/

[30] FDA, *Overview of Device Regulation*, Feb. 2019. [Online]. Available: http://www.fda.gov/medical-devices/device-advice-comprehensive-regulatory-assistance/overview-device-regulation

[31] Center for Devices and Radiological Health, "Cybersecurity for networked medical devices containing off-the-shelf (ots) software," January 2005. [Online]. Available: https://www.fda.gov/regulatory-information/search-fda-guidance-documents/cybersecurity-networked-medical-devices-containing-shelf-ots-software

[32] Food and Drug Administration, *General Principles of Software Validation*, May 2002. [Online]. Available: http://www.fda.gov/regulatory-information/search-fda-guidance-documents/general-principles-software-validation

[33] FDA, "Cybersecurity for Networked Medical Devices Containing Off-the-Shelf (OTS) Software," 2005. [Online]. Available: https://www.fda.gov/regulatory-information/search-fda-guidance-documents/cybersecurity-networked-medical-devices-containing-shelf-ots-software

[34] FDA Center for Devices and Radiological Health, "Postmarket Management of Cybersecurity in Medical Devices," Dec. 2016. [Online]. Available: http://www.fda.gov/regulatory-information/search-fda-guidance-documents/postmarket-management-cybersecurity-medical-devices

[35] N. Hrgarek, "Certification and regulatory challenges in medical device software development," in *2012 4th International Workshop on Software Engineering in Health Care (SEHC)*, Jun. 2012, pp. 40–43. [Online]. Available: https://ieeexplore.ieee.org/document/6227011

[36] C. Morgan, "Medical device cybersecurity regulatory publications," November 2019. [Online]. Available: https://www.apraciti.com/blog/2019/11/25/global-regulatory-authority-publications-on-medical-device-cybersecurity

[37] FDA, *Recognized Consensus Standards*. [Online]. Available: https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfStandards/results.cfm

[38] ——, "Cybersecurity Digital Health Initative," Jun. 2019. [Online]. Available: https://www.fda.gov/medical-devices/digital-health/cybersecurity

[39] FDA Center for Devices and Radiological Health, "Content of Premarket Submissions for Management of Cybersecurity in Medical Devices," Feb. 2019. [Online]. Available: http://www.fda.gov/regulatory-information/search-fda-guidance-documents/content-premarket-submissions-management-cybersecurity-medical-devices-0

[40] National Telecommunications and Information Administration, "Software component transparency: Healthcare proof of concept report," October 2019. [Online]. Available: https://www.ntia.doc.gov/SoftwareTransparency

[41] River Loop Security, "Reactions to fda draft cybersecurity guidance." [Online]. Available: https://www.riverloopsecurity.com/blog/2018/11/fda-guidance-nov2018/

[42] M. Weiss and H. Mohr, "Spinal-cord stimulators help some patients, injure others," November 2018. [Online]. Available: https://apnews.com/86ba45b0a4ad443fad1214622d13e6cb

[43] B. J. Carlson, "Why is an Ohio hospital getting false alerts from a Medtronic heart monitor?" [Online]. Available: http://www.startribune.com/why-is-an-ohio-hospital-getting-false-alerts-from-a-medtronic-heart-monitor/538070242/

[44] FDA, "Medtronic recalls remote controllers for minimed insulin pumps for potential cybersecurity risks," November 2019. [Online]. Available: https://www.fda.gov/medical-devices/medical-device-recalls/medtronic-recalls-remote-controllers-minimed-insulin-pumps-potential-cybersecurity-risks

[45] D. Klonoff and J. Han, "The first recall of a diabetes device because of cybersecurity risks." July 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/31313589

[46] R. Ross, M. McEvilley, and J. Oren, "Nist special publication 800-160: Systems security engineering considerations for a multidisciplinary approach in the engineering of trustworthy secure systems," *Gaithersburg: National Institute of Standards and Technology*, 2016. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-160/vol-1/final

[47] L. Coppolino, S. D'Antonio, G. Mazzeo, and L. Romano, "A comprehensive survey of hardware-assisted security: From the edge to the cloud," *Internet of Things*, vol. 6, p. 100055, Jun. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2542660519300101

[48] A. Shameli-Sendi, R. Aghababaei-Barzegar, and M. Cheriet, "Taxonomy of information security risk assessment (isra)," *Computers & Security*, vol. 57, pp. 14 – 30, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404815001650

[49] FIPS, PUB, "200," *Minimum Security Requirements for Federal Information and Information Systems*, vol. 2, 2006. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.200.pdf

[50] R. Shirey, "RFC 2828: Internet security glossary," *The Internet Society*, vol. 13, 2000. [Online]. Available: https://tools.ietf.org/html/rfc2828

[51] D. Kreutz, F. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," in *Proceedings of the second ACM SIGCOMM workshop on Hot*

topics in software defined networking. ACM, 2013, pp. 55–60. [Online]. Available: https://datatracker.ietf.org/meeting/87/materials/slides-87-sdnrg-2

[52] W. A. Jansen, "Countermeasures for mobile agent security," *Computer communications*, vol. 23, no. 17, pp. 1667–1676, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S014036640000253X?via%3Dihub

[53] A. Goldstein and U. Frank, "Components of a multi-perspective modeling method for designing and managing it security systems," *Information Systems and e-Business Management*, vol. 14, no. 1, pp. 101–140, 2016. [Online]. Available: https://www.researchgate.net/publication/273258830_Components_of_a_Multi-Perspective_Modeling_Method_for_Designing_and_Managing_IT-Security_Systems

[54] F. Swiderski and W. Snyder, *Threat modeling*. Microsoft Press, 2004. [Online]. Available: https://dl.acm.org/doi/book/10.5555/983226

[55] G. McGraw, B. Chess, and S. Migues, "Building security in maturity model," *Fortify & Cigital*, 2009. [Online]. Available: https://www.owasp.org/images/b/bd/Bsimm09.pdf

[56] C. Camara, P. Peris-Lopez, and J. E. Tapiador, "Security and privacy issues in implantable medical devices: A comprehensive survey," *Journal of Biomedical Informatics*, vol. 55, pp. 272–289, Jun. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S153204641500074X

[57] R. Altawy and A. M. Youssef, "Security Tradeoffs in Cyber Physical Systems: A Case Study Survey on Implantable Medical Devices," *IEEE Access*, vol. 4, pp. 959–979, 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7393449

[58] F. Loi, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, "Systematically Evaluating Security and Privacy for Consumer IoT Devices," in *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy - IoTS&P '17*. Dallas, Texas, USA: ACM Press, 2017, pp. 1–6. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3139937.3139938

[59] B. Alexander, S. Haseeb, and A. Baranchuk, "Are implanted electronic devices hackable?" *Trends in Cardiovascular Medicine*, Nov. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1050173818302597

[60] B. M. Kuehn, "Pacemaker recall highlights security concerns for implantable devices," 2018. [Online]. Available: https://www.ahajournals.org/doi/full/10.1161/CIRCULATIONAHA.118.037331

[61] J. Davis, "82% iot devices of health providers, vendors targeted by cyberattacks," August 2019. [Online]. Available: https://healthitsecurity.com/news/82-iot-devices-of-health-providers-vendors-targeted-by-cyberattacks

[62] irdeto, "Irdeto global connected industries cybersecurity survey," August 2019. [Online]. Available: https://go.irdeto.com/connected-industries-cybersecurity-survey-report/

[63] Z. Doffman, "Cyberattacks On IOT Devices Surge 300% In 2019, 'Measured In Billions', Report Claims," September 2019. [Online]. Available: https://www.forbes.com/sites/zakdoffman/2019/09/14/dangerous-cyberattacks-on-iot-devices-up-300-in-2019-now-rampant-report-claims/#2998b0145892

[64] M. Michael, "Attack landscape h1 2019: Iot, smb traffic abound," September 2019. [Online]. Available: https://blog.f-secure.com/attack-landscape-h1-2019-iot-smb-traffic-abound/

[65] B. Stack, "Here's how much your personal information is selling for on the dark web," December 2017. [Online]. Available: https://www.experian.com/blogs/ask-experian/heres-how-much-your-personal-information-is-selling-for-on-the-dark-web/

[66] FireEye, "Beyond Compliance: Cyber Threats and Healthcare," Tech. Rep., 2019. [Online]. Available: https://www.fireeye.com/blog/threat-research/2019/08/healthcare-research-data-pii-continuously-targeted-by-multiple-threat-actors.html

[67] T. Brewster, "Medical devices hit by ransomware for the first time in us hospitals," May 2017. [Online]. Available: https://www.forbes.com/sites/thomasbrewster/2017/05/17/wannacry-ransomware-hit-real-medical-devices/#32a8af25425c

[68] MITRE, "Apt18." [Online]. Available: https://attack.mitre.org/groups/G0026/

[69] MITRE, "Apt41." [Online]. Available: https://attack.mitre.org/groups/G0096/

[70] Symantec, "New orangeworm attack group targets the healthcare sector in the u.s., europe, and asia," April 2018. [Online]. Available: https://www.symantec.com/blogs/threat-intelligence/orangeworm-targets-healthcare-us-europe-asia

[71] Y. Mirsky, T. Mahler, I. Shelef, and Y. Elovici, "Ct-gan: Malicious tampering of 3d medical imagery using deep learning," *arXiv preprint arXiv:1901.03597*, 2019. [Online]. Available: https://arxiv.org/abs/1901.03597

[72] C. Beek, "Mcafee researchers find poor security exposes medical data to cybercriminals." [Online]. Available: https://securingtomorrow.mcafee.com/other-blogs/mcafee-labs/mcafee-researchers-find-poor-security-exposes-medical-data-to-cybercriminals/

[73] J. Umawing, "Sophisticated threats plague ailing healthcare industry." [Online]. Available: https://blog.malwarebytes.com/cybercrime/2019/04/sophisticated-threats-plague-ailing-healthcare-industry/

[74] L. H. Newman, "A New Pacemaker Hack Puts Malware Directly on the Device," *Wired*, Aug. 2018. [Online]. Available: https://www.wired.com/story/pacemaker-hack-malware-black-hat/

[75] Y.-H. Joung, "Development of implantable medical devices: from an engineering perspective," *International Neurourology Journal*, vol. 17, no. 3, p. 98, 2013. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/24143287

[76] S. Khandelwal, "Over 8,600 vulnerabilities found in pacemakers," Swati Khandelwal 2017. [Online]. Available: https://thehackernews.com/2017/06/pacemaker-vulnerability.html

[77] B. Rios and J. Butts, "Security evaluation of the implantable cardiac device ecosystem architecture and implementation interdependencies," WhiteScope, Tech. Rep., May 2017. [Online]. Available: https://drive.google.com/file/d/0B_GspGER4QQTYkJfaVlBeGVCSW8/view

[78] Billy Rios and Jonathan Butts. Understanding pacemaker systems cybersecurity. Whitescope. [Online]. Available: http://blog.whitescope.io/2017/05/understanding-pacemaker-systems.html

[79] M. Rushanan, A. D. Rubin, D. F. Kune, and C. M. Swanson, "Sok: Security and privacy in implantable medical devices and body area networks," in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 524–539. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6956585

[80] M. Rostami, A. Juels, and F. Koushanfar, "Heart-to-heart (h2h): authentication for implanted medical devices," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 1099–1112. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/2508859.2516658?download=true

[81] S. Zhang, *People Are Clamoring to Buy Old Insulin Pumps*, Apr. 2019. [Online]. Available: https://www.theatlantic.com/science/archive/2019/04/looping-created-insulin-pump-underground-market/588091/

[82] M. E. Whitman, "Enemy at the gate: threats to information security," *Communications of the ACM*, vol. 46, no. 8, p. 91, 2003. [Online]. Available: https://dl.acm.org/doi/10.1145/859670.859675

[83] ISO 14971:2007, *Medical devices – Application of risk management to medical devices*. [Online]. Available: http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/03/81/38193.html

[84] I. Stine, M. Rice, S. Dunlap, and J. Pecarina, "A cyber risk scoring system for medical devices," *International Journal of Critical Infrastructure Protection*, vol. 19, pp. 32–46, Dec. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S187454821730063X

[85] Healthcare Information and Management Systems Society (HIMSS), "Manufacturer disclosure statement for medical device security (mds2)." [Online]. Available: https://www.himss.org/resourcelibrary/MDS2

[86] S. Bhunia and M. Tehranipoor, "Chapter 2 - A Quick Overview of Electronic Hardware," in *Hardware Security*, S. Bhunia and M. Tehranipoor, Eds. Morgan Kaufmann, Jan. 2019, pp. 23–45. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128124772000071

[87] FDA Office of the Commissioner, "Statement from FDA Commissioner Scott Gottlieb, M.D. on FDA's efforts to strengthen the agency's medical device cybersecurity program as part of its mission to protect patients," Feb. 2019. [Online]. Available: https://www.fda.gov/news-events/press-announcements/statement-fda-commissioner-scott-gottlieb-md-fdas-efforts-strengthen-agencys-medical-device

[88] Center for Devices and Radiological Health, "Digital Health," Jul. 2019. [Online]. Available: http://www.fda.gov/medical-devices/digital-health

[89] TC 62/SC 62A, "IEC 62304:2006+AMD1:2015 CSV Consolidated version, Medical device software - Software life cycle processes," Tech. Rep., Jun. 2015. [Online]. Available: https://webstore.iec.ch/publication/22794

[90] AAMI, *ANSI/AAMI SW91:2018 (PDF)*. [Online]. Available: https://my.aami.org/store/detail.aspx?id=SW912018PDF

[91] A. Ray, R. Jetley, and P. Jones, "Engineering high confidence medical device software," *ACM SIGBED Review*, vol. 6, no. 2, pp. 1–7, Jul. 2009. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1859823.1859824

[92] NITRD, "High-Confidence Medical Devices: Cyber-Physical Systems for 21st Century Health Care," Tech. Rep., 2009. [Online]. Available: https://www.nitrd.gov/About/MedDevice-FINAL1-web.pdf

[93] Insup Lee, G. Pappas, R. Cleaveland, J. Hatcliff, B. Krogh, P. Lee, H. Rubin, and Lui Sha, "High-Confidence Medical Device Software and Systems," *Computer*, vol. 39, no. 4, pp. 33–38, Apr. 2006. [Online]. Available: http://ieeexplore.ieee.org/document/1620992/

[94] Diabetes Technology Society Standard for Wireless Device Security (DTSec) working group, "DTSec Protection Profile," Common Criteria Protection Profile Version 2.0, Nov. 2017. [Online]. Available: https://www.diabetestechnology.org/dtsec/DTSec%20PP.pdf

[95] ——, "Protection Profile for Connected Diabetes Devices (CDD PP) Extended Package: Enhanced Basic," Tech. Rep., May 2018. [Online]. Available: https://www.diabetestechnology.org/dtmost/DTMoST%20Enhanced%20Basic%20Extended%20Package.pdf

[96] ——, "Protection Profile for Connected Diabetes Devices (CDD PP) Extended Package: Moderate," Tech. Rep., May 2018. [Online]. Available: https://www.diabetestechnology.org/dtmost/DTMoST%20Moderate%20Extended%20Package.pdf

[97] A. Baranchuk, M. M. Refaat, K. K. Patton, M. K. Chung, K. Krishnan, V. Kutyifa, G. Upadhyay, J. D. Fisher, D. R. Lakkireddy, and , "Cybersecurity for cardiac implantable electronic devices," *Journal of the American College of Cardiology*, vol. 71, no. 11, pp. 1284–1288, 2018. [Online]. Available: http://www.onlinejacc.org/content/71/11/1284

[98] Health Care Industry Cybersecurity Task Force, "Report On Improving Cybersecurity In The Health Care Industry," June 2017. [Online]. Available: https://www.phe.gov/preparedness/planning/cybertf/documents/report2017.pdf

[99] Healthcare & Public Health Sector Coordinating Councils, "Health industry cybersecurity practices (hicp): Managing threats and protecting patients," May 2018. [Online]. Available: https://healthsectorcouncil.org/wp-content/uploads/2018/12/HICP-Main-508.pdf

[100] ——, "Medical device and healthcare information technology joint security plan," January 2019. [Online]. Available: https://healthsectorcouncil.org/the-joint-security-plan/

[101] Booz Allen Hamilton and the eHealth Initiative, "Securing connected medical devices," October 2019. [Online]. Available: https://www.ehidc.org/sites/default/files/resources/files/Securing%20Connected%20Medical%20Devices%20FINAL%2010.28.19.pdf

[102] H. Elmiligi, F. Gebali, and M. W. El-Kharashi, "Multi-dimensional analysis of embedded systems security," *Microprocessors and Microsystems*, vol. 41, pp. 29–36, Mar. 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0141933116000028

[103] F. M. Tabrizi and K. Pattabiraman, "Design-Level and Code-Level Security Analysis of IoT Devices," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 3, pp. 1–25, May 2019. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3323876.3310353

[104] J. Chaudhry, C. Valli, M. Crowley, J. Haass, and P. Roberts, "POStCODE Middleware for Post-Market Surveillance of Medical Devices for Cyber Security in Medical and Healthcare Sector in Australia," in *2018 12th International Symposium on Medical Information and Communication Technology (ISMICT)*, Mar. 2018, pp. 1–10. [Online]. Available: https://ieeexplore.ieee.org/document/8573695

[105] M. Zalewski, "american fuzzy lop." [Online]. Available: http://lcamtuf.coredump.cx/afl/

[106] A. Nguyen-Tuong, D. Melski, J. W. Davidson, M. Co, W. Hawkins, J. D. Hiser, D. Morris, D. Nguyen, and E. Rizzi, "Xandra: An Autonomous Cyber Battle System for the Cyber Grand Challenge," *IEEE Security Privacy*, vol. 16, no. 2, pp. 42–51, Mar. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8328984

[107] J. Zaddach, L. Bruno, A. Francillon, and D. Balzarotti, "Avatar: A Framework to Support Dynamic Security Analysis of Embedded Systems' Firmwares," in *Proceedings 2014 Network and Distributed System Security Symposium*. San Diego, CA: Internet Society, 2014. [Online]. Available: http://www.syssec-project.eu/m/page-media/3/ndss14_zaddach.pdf

[108] H. K. Rajaram, "Taxonomy Based Testing Using SW91, a Medical Device Software Defect Taxonomy," in *2018 IEEE 11th International Conference on Software Testing, Verification and Validation (ICST)*, Apr. 2018, pp. 422–423. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8367071

[109] MITRE, *CWE - CWE-884: CWE Cross-section (3.3)*. [Online]. Available: https://cwe.mitre.org/data/definitions/884.html

[110] H. K. Rajaram, J. Loane, S. T. MacMahon, and F. M. Caffery, "Taxonomy-based testing and validation of a new defect classification for health software," *Journal of Software: Evolution and Process*, vol. 31, no. 1, p. e1985, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1985

[111] A. Al-Omary, H. M. AlSabbagh, and H. Al-Rizzo, "Survey of Hardware-based Security support for IoT/CPS Systems," *KnE Engineering*, vol. 3, no. 7, p. 52, Oct. 2018. [Online]. Available: https://knepublishing.com/index.php/KnE-Engineering/article/view/3072

[112] R. Spring, E. Freudenthal, and L. Estevez, "Practical techniques for limiting disclosure of RF-equipped medical devices," in *2007 IEEE Dallas Engineering in Medicine and Biology Workshop*. IEEE, 2007, pp. 82–85. [Online]. Available: https://ieeexplore.ieee.org/document/4454179

[113] D. Halperin, T. S. Heydt-Benjamin, B. Ransford, S. S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. H. Maisel, "Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE, 2008, pp. 129–142. [Online]. Available: https://ieeexplore.ieee.org/document/4531149

[114] N. Ellouze, S. Rekhis, N. Boudriga, and M. Allouche, "Powerless security for Cardiac Implantable Medical Devices: Use of Wireless Identification and Sensing Platform," *Journal of Network and Computer Applications*, vol. 107, pp. 1–21, Apr. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804518300237

[115] T. Denning, K. Fu, and T. Kohno, "Absence makes the heart grow fonder: New directions for implantable medical device security." in *HotSec*, 2008. [Online]. Available: https://pdfs.semanticscholar.org/641b/9d0b1733e2c3a6989d6e555c9150432c56c1.pdf

[116] K. B. Rasmussen, C. Castelluccia, T. S. Heydt-Benjamin, and S. Capkun, "Proximity-based access control for implantable medical devices," in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 410–419. [Online]. Available: https://www.researchgate.net/publication/221609745_Proximity-based_access_control_for_implantable_medical_devices

[117] S. Schechter, "Security that is Meant to be Skin Deep Using Ultraviolet Micropigmentation to Store Emergency-Access Keys for Implantable Medical Devices," 2010. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/healthsec-2.pdf

[118] T. Denning, A. Borning, B. Friedman, B. T. Gill, T. Kohno, and W. H. Maisel, "Patients, pacemakers, and implantable defibrillators: Human values and security for wireless implantable medical devices," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2010, pp. 917–926. [Online]. Available: https://dl.acm.org/doi/10.1145/1753326.1753462

[119] X. Hei, X. Du, J. Wu, and F. Hu, "Defending resource depletion attacks on implantable medical devices," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*. IEEE, 2010, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5685228

[120] F. Xu, Z. Qin, C. C. Tan, B. Wang, and Q. Li, "Imdguard: Securing implantable medical devices with the external wearable guardian," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 1862–1870.

[121] C. Li, A. Raghunathan, and N. K. Jha, "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*. IEEE, 2011, pp. 150–156. [Online]. Available: https://ieeexplore.ieee.org/document/6026732

[122] S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, and K. Fu, "They can hear your heartbeats: non-invasive security for implantable medical devices," in *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4. ACM, 2011, pp. 2–13. [Online]. Available: https://dl.acm.org/doi/10.1145/2018436.2018438

[123] S. Hosseini-Khayat, "A lightweight security protocol for ultra-low power asic implementation for wireless implantable medical devices," in *2011 5th International Symposium on Medical Information and Communication Technology*. IEEE, 2011, pp. 6–9. [Online]. Available: https://ieeexplore.ieee.org/document/5759785

[124] A. Kaadan and H. H. Refai, "Securing wireless medical devices," in *2012 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2012, pp. 942–948. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6503234

[125] V. Pournaghshband, M. Sarrafzadeh, and P. Reiher, "Securing legacy mobile medical devices," in *International Conference on Wireless Mobile Communication and Healthcare*. Springer, 2012, pp. 163–172. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-37893-5_19

[126] D. F. Kune, J. Backes, S. S. Clark, D. Kramer, M. Reynolds, K. Fu, Y. Kim, and W. Xu, "Ghost talk: Mitigating emi signal injection attacks against analog sensors," in *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013, pp. 145–159. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6547107

[127] C. Strydis, R. M. Seepers, P. Peris-Lopez, D. Siskos, and I. Sourdis, "A system architecture, processor, and communication protocol for secure implants," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 10, no. 4, p. 57, 2013. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/2541228.2555313?download=true

[128] N. Henry, N. Paul, and N. McFarlane, "Using bowel sounds to create a forensically-aware insulin pump system," in *Presented as part of the 2013 {USENIX} Workshop on Health Information Technologies*, 2013. [Online]. Available: https://www.usenix.org/system/files/conference/healthtech13/healthtech13-henry.pdf

[129] M. Zhang, A. Raghunathan, and N. K. Jha, "MedMon: Securing Medical Devices Through Wireless Monitoring and Anomaly Detection," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 7, no. 6, pp. 871–881, Dec. 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6507636/

[130] M. Darji and B. H. Trivedi, "Detection of active attacks on wireless imds using proxy device and localization information," in *International Symposium on Security in Computing and Communication*. Springer, 2014, pp. 353–362. [Online]. Available: https://www.semanticscholar.org/paper/Detection-of-Active-Attacks-on-Wireless-IMDs-Using-Darji-Trivedi/7e0b17c4d898d03ea7867af39d45dcdfd0138e16

[131] X. Hei, X. Du, S. Lin, I. Lee, and O. Sokolsky, "Patient infusion pattern based access control schemes for wireless insulin pump system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 11, pp. 3108–3121, 2014. [Online]. Available: http://www.ece.sunysb.edu/~slin/Publications/comsoc_main.pdf

[132] L. Shi, M. Li, S. Yu, and J. Yuan, "BANA: body area network authentication exploiting channel characteristics," *IEEE Journal on selected Areas in Communications*, vol. 31, no. 9, pp. 1803–1816, 2013. [Online]. Available: https://www.semanticscholar.org/paper/BANA%3A-Body-Area-Network-Authentication-Exploiting-Shi-Li/256249379a263953e70232fa8d3b7463a156d800

[133] B. Kim, J. Yu, and H. Kim, "In-vivo NFC: Remote monitoring of implanted medical devices with improved privacy," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 2012, pp. 327–328. [Online]. Available: https://www.semanticscholar.org/paper/In-vivo-NFC%3A-remote-monitoring-of-implanted-medical-Kim-Yu/7f36e3ff06dc785dc312481a2b52935500d2fe84

[134] C.-S. Park, "Security mechanism based on hospital authentication server for secure application of implantable medical devices," *BioMed research international*, vol. 2014, 2014. [Online]. Available: https://www.hindawi.com/journals/bmri/2014/543051/

[135] David DeBelser and Michael L. Blomquist, "Security features for a medical infusion pump," Patent, Apr., 2009. [Online]. Available: https://patents.google.com/patent/US9192712B2/en

[136] D. Arora, S. Ravi, A. Raghunathan, and N. K. Jha, "Hardware-Assisted Run-Time Monitoring for Secure Program Execution on Embedded Processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 12, pp. 1295–1308, Dec. 2006. [Online]. Available: http://ieeexplore.ieee.org/document/4052340/

[137] A. Tang, S. Sethumadhavan, and S. Stolfo, "Heisenbyte: Thwarting Memory Disclosure Attacks Using Destructive Code Reads," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 256–267. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813685

[138] A. Cui and S. J. Stolfo, "Defending Embedded Systems with Software Symbiotes," in *Recent Advances in Intrusion Detection*, ser. Lecture Notes in Computer Science, R. Sommer, D. Balzarotti, and G. Maier, Eds. Springer Berlin Heidelberg, 2011, pp. 358–377. [Online]. Available: http://nsl.cs.columbia.edu/projects/minestrone/papers/Symbiotes.pdf

[139] S. S. Clark, B. Ransford, A. Rahmati, S. Guineau, J. Sorber, W. Xu, and K. Fu, "WattsUpDoc: Power Side Channels to Nonintrusively Discover Untargeted Malware on Embedded Medical Devices," 2013. [Online]. Available: https://www.usenix.org/conference/healthtech13/workshop-program/presentation/Clark

[140] J.-W. Hu, L.-Y. Yeh, S.-W. Liao, and C.-S. Yang, "Autonomous and malware-proof blockchain-based firmware update platform with efficient batch verification for Internet of Things devices," *Computers & Security*, vol. 86, pp. 238–252, Sep. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016740481831438X

[141] I. Wagner, V. Bertacco, and T. Austin, "Shielding against design flaws with field repairable control logic," in *Proceedings of the 43rd annual conference on Design automation - DAC '06*. San Francisco, CA, USA: ACM Press, 2006, p. 344. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1146909.1146998

[142] Y. Cao, C. Hu, B. Ma, H. Hao, L. Li, and W. Wang, "Secure Method for Software Upgrades for Implantable Medical Devices," *Tsinghua Science & Technology*, vol. 15, no. 5, pp. 517–525, Oct. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S100702141070095X

[143] M. L. Hale, K. Lotfy, R. F. Gamble, C. Walter, and J. Lin, "Developing a platform to evaluate and assess the security of wearable devices," *Digital Communications and Networks*, Oct. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352864817302985

[144] T. S. Hoang, C. Snook, A. Salehi, M. Butler, and L. Ladenberger, "Validating and verifying the requirements and design of a haemodialysis machine using the Rodin toolset," *Science of Computer Programming*, vol. 158, pp. 122–147, Jun. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167642317302381

[145] P. Arcaini, S. Bonfanti, A. Gargantini, A. Mashkoor, and E. Riccobene, "Integrating formal methods into medical software development: The ASM approach," *Science of Computer Programming*, vol. 158, pp. 148–167, Jun. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167642317301430

[146] H. M. Conboy, G. S. Avrunin, and L. A. Clarke, "Process-based derivation of requirements for medical devices," in *Proceedings of the ACM international conference on Health informatics - IHI '10*. Arlington, Virginia, USA: ACM Press, 2010, p. 656. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1882992.1883095

[147] M. Lepmets, F. Mc Caffery, and P. Clarke, "Piloting MDevSPICE: the medical device software process assessment framework," in *Proceedings of the 2015 International Conference on Software and System Process - ICSSP 2015*. Tallinn, Estonia: ACM Press, 2015, pp. 9–16. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2785592.2785598

[148] S. Bonfanti, A. Gargantini, and A. Mashkoor, "A systematic literature review of the use of formal methods in medical software systems," *Journal of Software: Evolution and Process*, vol. 30, no. 5, p. e1943, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1943

[149] M. Pajic, Z. Jiang, A. Connolly, S. Dixit, and R. Mangharam, "A platform for implantable medical device validation," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks - IPSN '10*. Stockholm, Sweden: ACM Press, 2010, p. 418. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1791212.1791284

[150] J. Bowen and S. Reeves, "Modelling safety properties of interactive medical systems," in *Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems - EICS '13*. London, United Kingdom: ACM Press, 2013, p. 91. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2494603.2480314

[151] M. D. Harrison, L. Freitas, M. Drinnan, J. C. Campos, P. Masci, C. di Maria, and M. Whitaker, "Formal techniques in the safety analysis of software components of a new dialysis machine," *Science of Computer Programming*, vol. 175, pp. 17–34, Apr. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167642318300819

[152] T. Nguyen, W. Weimer, C. L. Goues, and S. Forrest, "Using Execution Paths to Evolve Software Patches," in *2009 International Conference on Software Testing, Verification, and Validation Workshops*, Apr. 2009, pp. 152–153. [Online]. Available: https://ieeexplore.ieee.org/document/4976381

[153] Z. Qi, F. Long, S. Achour, and M. Rinard, "An analysis of patch plausibility and correctness for generate-and-validate patch generation systems," in *Proceedings of the 2015 International Symposium on Software Testing and Analysis - ISSTA 2015*. Baltimore, MD, USA: ACM Press, 2015, pp. 24–36. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2771783.2771791

[154] Y. Xiong, X. Liu, M. Zeng, L. Zhang, and G. Huang, "Identifying patch correctness in test-based program repair," in *Proceedings of the 40th International Conference on Software Engineering - ICSE '18*. Gothenburg, Sweden: ACM Press, 2018, pp. 789–799. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3180155.3180182

[155] W. H. Hawkins, J. D. Hiser, M. Co, A. Nguyen-Tuong, and J. W. Davidson, "Zipr: Efficient Static Binary Rewriting for Security," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Jun. 2017, pp. 559–566. [Online]. Available: https://ieeexplore.ieee.org/document/8023154

[156] J. Avery and J. R. Wallrabenstein, "Formally modeling deceptive patches using a game-based approach," *Computers & Security*, vol. 75, pp. 182–190, Jun. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404818301330

[157] S. Allen, "Medical device software under the microscope," *Network Security*, vol. 2014, no. 2, pp. 11–12, Feb. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1353485814700212

[158] A. Murugesan, O. Sokolsky, S. Rayadurgam, M. Whalen, M. Heimdahl, and I. Lee, "Linking abstract analysis to concrete design: A hierarchical approach to verify medical CPS safety," in *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. Berlin, Germany: IEEE, Apr. 2014, pp. 139–150. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6843718

[159] Muthukumar N., S. Srinivasan, K. Ramkumar, D. Pal, J. Vain, and S. Ramaswamy, "A model-based approach for design and verification of Industrial Internet of Things," *Future Generation Computer Systems*, vol. 95, pp. 354–363, Jun. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X18321137

[160] L. H. Newman, "A Model Hospital Where the Devices Get Hacked—on Purpose," *Wired*, Aug. 2019. [Online]. Available: https://www.wired.com/story/defcon-medical-device-village-hacking-hospital/

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

[162] C. Ngak, "Black hat hacker can remotely attack insulin pumps and kill people," August 2011. [Online]. Available: https://www.cbsnews.com/news/black-hat-hacker-can-remotely-attack-insulin-pumps-and-kill-people/

[163] I. Onu, "Hackers Almost Killed a Child in a Tyumen Hospital," July 2018. [Online]. Available: http://www.themicrople.com/news/hackers-sberbank-tyumen-hospital/

[164] A. Kut, "Hackers attacked a Russian hospital during surgery on the brain of the child," June 2018. [Online]. Available: https://handofmoscow.com/2018/07/06/hackers-attacked-a-russian-hospital-during-surgery-on-the-brain-of-the-child/

[165] C. Summit, "Sim runaway pacemaker from the cybermed summit," January 2020. [Online]. Available: https://vimeo.com/390057154

[166] R. Conference, "Live simulation: A medical device hack and patient codes — rsac 2018," July 2018. [Online]. Available: https://youtu.be/OpyYLJOLwpA

[167] "The cybermed summit." [Online]. Available: https://www.cybermedsummit.org/

[168] A. Biondo, "Coverage-guided fuzzing of embedded firmware with avatar2," Master's thesis, Università degli Studi di Padova, 2019. [Online]. Available: https://siagas.math.unipd.it/siagas/getTesi.php?id=2030

[169] Y. Zheng, A. Davanian, H. Yin, C. Song, H. Zhu, and L. Sun, "Firm-afl: High-throughput greybox fuzzing of iot firmware via augmented process emulation," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 1099–1114. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/zheng

[170] D. Antonioli, N. O. Tippenhauer, and K. B. Rasmussen, "The KNOB is Broken: Exploiting Low Entropy in the Encryption Key Negotiation Of Bluetooth BR/EDR," in *USENIX*. Santa Clara, CA: USENIX Association, August 2019, pp. 104–1061. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/antonioli

[171] Center for Devices and Radiological Health, "Content of Premarket Submissions for Management of Cybersecurity in Medical Devices: for Management of Cybersecurity in Medical Devices Draft Guidance for Industry and Food and Drug Administration Staff [DRAFT GUIDANCE]," Oct 2018. [Online]. Available: http://www.fda.gov/regulatory-information/search-fda-guidance-documents/content-premarket-submissions-management-cybersecurity-medical-devices

[172] W. O. Redwood, "Cyber Physical System Vulnerability Research," Ph.D. dissertation, Florida State University, 2015. [Online]. Available: https://fsu.digital.flvc.org/islandora/object/fsu%3A360429

[173] M. Muench, D. Nisi, A. Francillon, and D. Balzarotti, "Avatar^2: A multi-target orchestration platform," in *BAR 18*, San Diego (USA). [Online]. Available: http://s3.eurecom.fr/docs/bar18_muench.pdf

[174] S. Nema, "Symbiotic design for cyber physical systems." [Online]. Available: https://www.darpa.mil/program/symbiotic-design-for-cyber-physical-systems

[175] J. Sanders, "Docker containers are filled with vulnerabilities: Here's how the top 1,000 fared," June 2019. [Online]. Available: https://www.techrepublic.com/article/docker-containers-are-filled-with-vulnerabilities-heres-how-the-top-1000-fared/

[176] DARPA, "Neurotechnology provides near-natural sense of touch," Sept 11, 2015. [Online]. Available: https://www.darpa.mil/news-events/2015-09-11

[177] E. Musk, "An integrated brain-machine interface platform with thousands of channels," *(Preprint) bioRxiv*, Aug 2, 2019. [Online]. Available: https://www.biorxiv.org/content/10.1101/703801v4

[178] WhiteScope, "Whitescope – adolus." [Online]. Available: https://www.adolus.com/whitescope-users/

[179] Firmadyne, "Firmadyne: Platform for emulation and dynamic analysis of Linux-based firmware," 2020. [Online]. Available: https://github.com/firmadyne/firmadyne

[180] C. Roberts, "Firmware slap," August 2019. [Online]. Available: https://github.com/ChrisTheCoolHut/Firmware_Slap

[181] Christopher Roberts. Firmware slap. DEF CON. [Online]. Available: https://media.defcon.org/DEF%20CON%2027/DEF%20CON%2027%20presentations/DEFCON-27-Christopher-Roberts-Firmware-Slap.pdf

[182] panda re, "PANDA: Platform for Architecture-Neutral Dynamic Analysis," 2020. [Online]. Available: https://github.com/panda-re/panda

[183] "Unicorn: The ultimate cpu emulator." [Online]. Available: https://www.unicorn-engine.org/

[184] trailofbits, "deepstate: A unit test-like interface for fuzzing and symbolic execution." [Online]. Available: https://github.com/trailofbits/deepstate

[185] ReFirmLabs, "binwalk: Firmware analysis tool." [Online]. Available: https://github.com/ReFirmLabs/binwalk

[186] "angr: python framework for analyzing binaries." [Online]. Available: https://angr.io/

[187] trailofbits, "grr: High-throughput fuzzer and emulator of DECREE binaries." [Online]. Available: https://github.com/trailofbits/grr

[188] Cyberhaven, "S^2E: A platform for in-vivo analysis of software systems." [Online]. Available: http://s2e.systems/

[189] L. C. Project, "libfuzzer – a library for coverage-guided fuzz testing." [Online]. Available: https://llvm.org/docs/LibFuzzer.html

[190] feliam, "pysymemu." [Online]. Available: https://github.com/feliam/pysymemu