Understanding the Security of Interoperable Medical Devices using Attack Graphs

Curtis R. Taylor, Krishna Venkatasubramanian, Craig A. Shue
Department of Computer Science,
Worcester Polytechnic Institute
Worcester, MA, 01609
crtaylor@cs.wpi.edu, kven@wpi.edu, cshue@cs.wpi.edu

ABSTRACT

Medical device interoperability is an increasingly prevalent example of how computing and information technology will revolutionize and streamline medical care. The overarching goal of interoperable medical devices (IMDs) is increased safety, usability, decision support, and a decrease in false alarms and clinician cognitive workload. One aspect that has not been considered thus far is ensuring IMDs do not inadvertently harm patients in the presence of malicious adversaries. Security for medical devices has gained some traction in the recent years following some well-publicized attacks on individual devices, such as pacemakers and insulin pumps. This has resulted in solutions being proposed for securing these devices, usually in stand-alone mode. However, the introduction of interoperability makes medical devices increasingly connected and dependent on each other. Therefore, security attacks on IMDs becomes easier to mount and in a stealthy manner.

This work outlines our effort in understanding the threats faced by IMDs, an important first step in eventually designing secure interoperability architectures. In this regard, we present: (1) a detailed attack graph-based analysis of threats on a specific interoperability environment based on providing patients pain medication (PCA) under various levels of interoperability from data aggregation to fully closed-loop control, (2) a description of the mitigation approaches possible for each of class of attack vectors identified, and (3) lessons learned from this experience which can be leveraged for improving existing IMD architectures. Our analysis demonstrates that even if we use provably safe medical systems in an interoperable setting with a safe interoperability engine, the presence of malicious behavior may render the entire setup unsafe for the patients.

General Terms

Security, Interoperability, Medical Device Systems

Keywords

Interoperable Medical Devices, Security, PCA, Infusion Pump

1. INTRODUCTION

Medical systems are increasingly being connected to each other as a way to improve patient safety. The ability of medical devices to interoperate with one another has the potential to yield better performance, from reduced false alarms to automatic decision/diagnosis support and medication interaction checking in real-time [1]. This interoperability can improve patient outcomes by reducing the 95K - 195K errors committed in U.S. hospitals [8].

While there may be impediments to device manufacturers providing interoperability with their competitors' medical devices, such as a lack of data standards, alternative mechanisms are possible. In particular, a communication/middleware standard would allow heterogenous devices to communicate with one another. The *MD PnP Integrated Clinical Architecture* (ICE) is one result of such standardizing efforts [2]. Although there can be interoperability at many different granularities from technical (being able to exchange bytes) to conceptual (shared assumptions about the reality at a meaningful abstraction) [20, 22], the interoperability in the ICE standard is somewhere in between syntactic (data format of communication is standard) and semantic (the meaning of the data being exchanged is unambiguously defined) levels.

The goal of ICE is to enable *safe interoperability* between medical devices. Specifically, safety in this context is defined as ensuring the patient's health is not harmed in anyway by the use of the *interoperable medical devices* (IMDs). One important issue that is not addressed in this standard is security. Considering security for IMDs is necessary because: (1) they deal with patient information and laws, such as the Health Insurance Portability and Accountability Act (HIPAA) [10], mandate it, and (2) security attacks can have serious safety consequences for the IMDs. In particular, a malicious entity can now easily suppress legitimate information and introduce bogus information between the devices and the middleware, leading to untimely or unwanted actuation or loss of privacy. *Therefore, we contend that both security and safety need to be enforced among the IMD devices to ensure that the patient is not harmed in any circumstance.* Recent years have brought increased attention to security vulnerabilities in standalone medical devices [5, 6, 11, 12]. However, the introduction of interoperability makes medical devices increasingly connected and dependent on each other. Therefore, security attacks on IMDs becomes easier to mount, and in a stealthy manner.

There has been growing interest in security issues pertaining to medical data collection, data transfer and processing, and electronic medical health records [3,9,21]. Standardization efforts are also underway [4,15,16]. However, these are focused on the information exchange aspects of the overall coordinating infrastructure and do not necessarily focus on medical device issues. This focus makes these standardization efforts incomplete. The proper development of strong security solutions for IMDs is still an open research question.

To develop security solutions for IMDs, we need a good understanding of the various threats to an IMD setup. In this paper we present a detailed description of attacks on a specific interoperability scenario for patient controlled analgesia (PCA). In this regard, we consider various levels of interoperability from simple-cases where interoperability promotes data aggregation to fully-closed-loop control of the actuating devices. The *principal contributions* of this paper therefore are:

- An attack graph-based description of attacks on IMDs when considering the PCA interoperability scenario.
- A description of the general mitigation strategies for each class of the attacks that are possible on the IMDs.
- A description of lessons learned from our experience, which can be used to design the interoperability architecture in a security-conscious manner.

We chose this PCA scenario for our IMD case-study because it is responsible for a very large number of treatment errors in the hospital setting. One study estimated that there are anywhere between 600,000 to 2 million adverse events in U.S. hospitals every year related to PCA [19]. Our analysis demonstrates that even if we use provably safe medical systems in an interoperable setting with a safe interoperability engine, in the presence of malicious behavior renders the entire setup unsafe — potentially harm inducing — for the patients.

To the best of our knowledge this is the first systematic attack description for a common treatment scenario (i.e., pain management), which can be implemented with IMDs in a realistic setting. Our work demonstrates that security has profound consequences to the safety of medical device interoperability and the patients they are serving. It is not just enough to design IMDs to be able to handle failures and software errors in order to be safe, they have to be secured from a variety of threats as well.

The paper is organized as follows: Section 2 presents background information on interoperability architecture standards and potential deployment approaches. Section 3 presents our problem statement along with the system and trust model. Section 4 illustrates attacks on the system. Section 5 presents the lessons learned and Section 6 presents the related work. Finally, Section 7 concludes the paper and presents the future work.

2. BACKGROUND

Rather than wait for the devices from different manufacturers to organically evolve interoperability capabilities, the *MD PnP Integrated Clinical Architecture* (ICE) was created

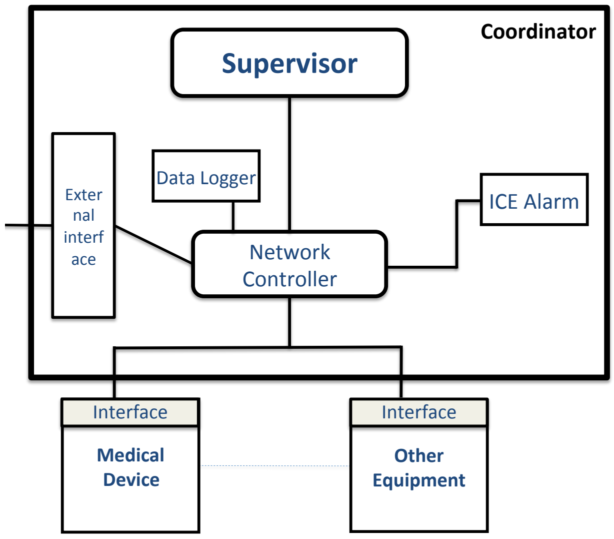


Figure 1: Interoperability architecture of MD PnP ICE standard

to enable diverse devices to talk to one another [2]. ICE was designed to act as a middleware to enable interaction of legacy, stand-alone medical devices and the applications using the medical devices. ICE also includes closed-loop physiological control, with automated sensing allowing automated actuation. The architecture of ICE typically consists of three entities:

- The *Medical Devices* and other devices on or around a single patient that can perform monitoring and actuation. The devices have an adapter that allows them to communicate with an entity called the Network Controller.
- The *Network Controller* interfaces with the medical devices. It is responsible for collecting data from the individual devices. It also connects the entire setup to an external network, such as the Healthcare Information System (HIS). The network controller also records all the actions of the entire system in a data logger for future analysis.
- The *Supervisor* receives data from the various medical devices, processes it, and initiates action from the medical devices. The Supervisor runs clinical applications that use the connected devices to support a clinical scenario selected by the caregiver.

The caregiver is responsible for configuring the system, selecting an appropriate program on the Supervisor, and then monitoring the patientÕs well-being using an user-interface provided by the Supervisor. The caregiver can control various parameters of the system, such as alarm thresholds, as permitted by the application the caregiver chooses (see Figure 1).

The entire system is designed to facilitate interaction between the medical devices available today. It has the potential to provide closed-loop control over the patient's health. For example, the Supervisor could run an application that receives data from a glucose monitor, processes the data to analyze the level of blood sugar in the patient, and commands the infusion pump to administer a dose of insulin chosen by the caregiver.

3. PROBLEM STATEMENT

Ensuring security is an important requirement for enabling the overall safety of the interoperable medical devices. Traditional solutions for building safe systems only considers "naturally-occurring" faults within the system. These do not include faults introduced into the system by adversaries, which may not follow the models of "naturally-occurring" faults, but instead act in unexpected ways. Hence, analyzing the security threats for interoperable medical devices is very important for ensuring that the IMDs are safe and do not harm the patient.

In this paper, we investigate the various ways in which a specific instantiation of interoperable medical devices can be attacked, in a systematic manner. We specifically consider cases where the individual devices are themselves "correct-by-design" and therefore are considered "safe" when they are operating in stand-alone fashion [17] . However, when malicious behavior is allowed, even such provably-safe devices working in conjunction with a safe and trusted coordinator in an interoperable environment, are inherently unsafe. We consider this analysis as a step towards building an effective architecture for securing interoperable medical devices. Before delving into the details of our security analysis, we present our system model and trust/threat model for this work.

3.1 System Model

Standards for interoperability between medical devices can support any combination of medical devices, provided they can be coordinated in a meaningful way to provide effective care for patients. The IMD configuration will vary to account for each patient's specific situation. In order to understand the security threats on IMDs, we consider a small IMD system, consisting of three devices, for a single patient needing pain management. As we will see, even in this very limited scenario, the avenues of attack are large and we can draw broad conclusions about IMD security threats.

Our scenario consists of an infusion pump programmed to infuse pain medication (e.g., morphine) to the patient as a specific (basal) rate. As pain medications tend to suppress respiration, we also have a pulse oximeter (measures level of O_2 in the blood) and a capnograph (measures CO_2 levels in the blood) to determine how the patient is responding to the pain medication. The measurement devices are collectively referred to as *sensors* in the rest of the paper. The infusion pump also allows the patient to press a bolus button to receive a single, large dose of the medication as needed. Obviously, frequent boluses should only be allowed for a patient if it is not suppressing their respiration to unhealthy levels.

All the medical devices in our setup interact with the coordinator. The details of the coordinator entity are abstracted out as our focus is primarily on its interaction with the medical devices. The coordinator is programmed by the caregiver by loading *medical applications* on it that perform specific tasks such as alarming or providing closed-loop control. In many instances, the coordinator can be used to control the individual medical devices. The coordinator has an internal alarm and logging capability and is connected to a patient display, which is a single entity and displays the patient's status in terms of the various physiological signals. The caregiver essentially monitors the patient through the patient display (dashed arrow in Figure 2). The coordinator also interfaces with the electronic health record (EHR) system. It can update and query the EHR when needed. For example, a medical application running on the coordinator can be used to perform a sanity check on the nurse's programming of the infusion pump based on medication orders in the EHR.

Our interoperability setup can be classified into four *configurations* based on level of control associated with the coordinator:

- *Simple (SC):* With a simple coordinator, the actuation and monitoring devices are programmed directly by the caregiver and then connected to the coordinator. The coordinator receives status updates from the individual medical devices, and it displays the information to the caregiver via the patient display. If the blood oxygen level of the patient goes below a certain threshold, a medical application on the coordinator will raise an alarm to the caregivers.

- *Alarming (AC):* In this scenario, the coordinator has the capability to program the devices as specified by the caregiver and monitor the patient's condition. If the blood oxygen level goes below a certain threshold, the coordinator (through a medical application executing on it) raises an alarm for the caregivers to react.

- *Bolus-controlling (BC):* In this scenario, the coordinator has the capability to program the devices as specified by the caregiver, raise an alarm if the patient's condition deteriorates, and control the frequency with which the patient can give themselves bolus doses.

- *Fully-closed Loop (FC):* In this scenario, the coordinator, after the initial programming by the caregiver, monitors the patient's condition, and if it deteriorates, automatically modifies the programming in place to reduce the safety risk, such as over-infusion, to the patient. Further, it can raise an alarm for the caregivers and also control the bolus volume and frequency for the patients.

Figure 2 (a), (b) shows the assumed interoperability setup for SC and the other modes (AC, BC and FC), respectively. The edges are labeled to indicate the information exchanged between the entities that the edge connects.

3.2 Trust Model

In our interoperability scenario, we consider the coordinator and the associated logging and alarms to be the only members of the trusted computing base (TCB). These components are trusted (they do not have malicious intent) and trustworthy (they will operate as expected). The dashed box in Figure 2 (a) and (b) signifies the TCB in our system model. Further, we assume that the caregiver is not necessarily trustworthy, in that the caregiver can make mistakes in programming the devices or may have malicious intent. We further assume that the infusion pump in our system model is verifiably safe as described by Kim *et al.* [17].

For our work, we essentially consider active adversaries (also called "attackers" interchangeably) who may interfere

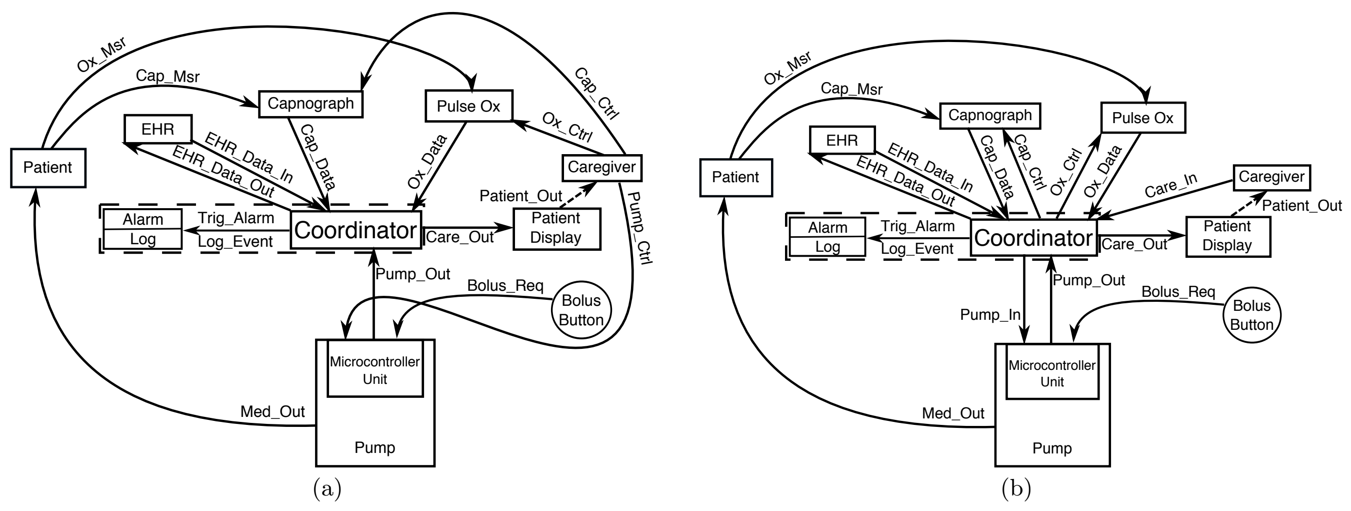


Figure 2: System Model for Interoperability Threat Analysis (a) Simple Case (SC), (b) Other cases (AC,BC, and FC)

with communication links, as per the Yao-Dolev model of an adversary [7]. The adversary may also physically alter the infusion pump or the line from the infusion pump to the patient, the coordinator, the pulse oximeter and capnograph. We assume that adversaries cannot modify the firmware of the devices, but the adversaries can mount limited physical attacks on the IMD setup. For example, the attacker can induce readings in the sensor and cut the infusion line to the patient. Note that, while adversaries may simply inject the patient directly and induce a medical emergency, we consider such attacks outside the scope of interoperable medical device security.

Finally, we only consider adversaries that induce over-infusion states (for pain medication under-infusion does not hamper patient safety) through the infusion pump. In other types of interoperability scenarios, both under-infusion and over-infusion can be problematic, such as with insulin infusion, which essentially doubles the threat surface.

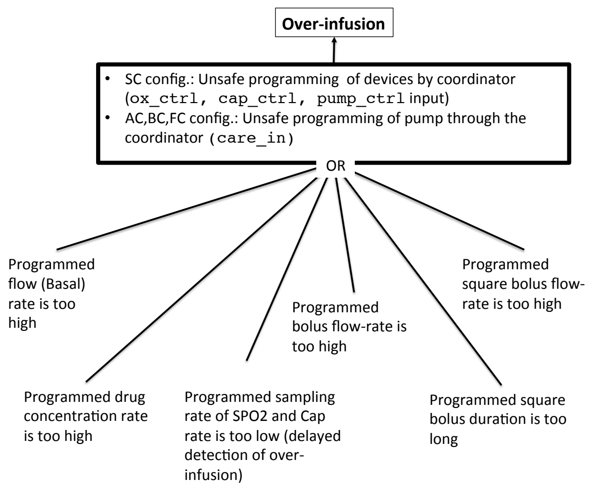
4. ANALYSIS OF ATTACKS ON INTEROPERABLE MEDICAL DEVICES

Before we can understand the security of IMDs, we must first examine their attack surfaces and the associated vulnerabilities. Importantly, by focusing on the assets to be protected and their associated vulnerabilities, we can determine remediation opportunities without having to anticipate an attacker's actions. In the context of medical devices, safety and security have a special relationship. The high-level patient safety goals vary dramatically based on a given patient scenario and set of devices. We consider an IMD scenario for patient-controlled analgesia (PCA), involving a PCA infusion pump, a pulse oximeter, and a capnograph, as a motivating example. The goal is to take a common treatment option in a hospital which can be improved using IMDs, evaluate its security in a systematic manner, and develop generalizable requirements for improving the safe operation of IMDs. In our scenario, there is one simple safety goal for the PCA pump: it must *not* administer an excessive quantity of pain medication (e.g., *over-infusion*). If this safety goal is violated, the patient's respiration may be suppressed. If not remediated, this may lead to patient mortality. In the remainder of this section, we focus on the vectors adversaries can use to subvert patient safety and harm the patient. We then discuss some viable countermeasures for these attacks.

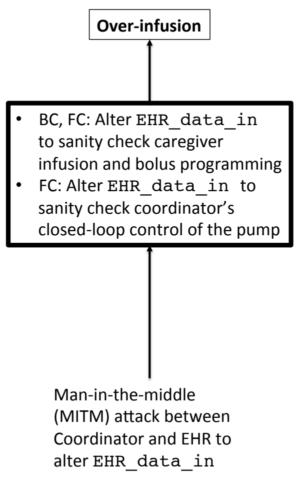
4.1 Attack Graphs

Over-infusion at the PCA pump is the only "unsafe" state for our case-study. If the infusion pump in our setup fails to infuse a sufficient quantity of analgesia, it is unlikely to cause a life-threatening event. Instead, the patient will experience pain and will alert a caregiver manually. However, in other scenarios, such as insulin infusion pumps, both the case of over-infusion and under-infusion can result in a safety violation. When considering the safety and security of IMDs, each unsafe state must be identified and the paths to that unsafe state enumerated. In Figures 3, 4, 5 we depict the *attack graphs* that describe various *attack vectors* that can lead to the over-infusion state for our setup. Each of these figures show the sub-branches of a larger attack tree for our scenario. Each of the attack vectors lead to the over-infusion attack, hence the root node for each of the trees in these figures is over-infusion. The figures are representing the following attack scenarios:

- Initialization Attacks: Represented in Figure 3 (a), these attack represents the situations where the caregiver programs the devices (using `cap_ctrl`, `ox_ctrl`, `pump_ctrl` in the SC case, and using `care_in` through the coordinator in the AC, BC, and FC cases) incorrectly.
- EHR Access Attacks: Represented in Figure 3 (b), these attacks represent the situations where the communication link between the coordinator and the EHR is compromised primarily through a man-in-the-middle attack.
- Partial Feedback Attacks: Represented in Figure 4 (a), these attacks represent the situations where the only some of feedback channels to the coordinator (e.g, `pump_out`, `ox_data` etc.). Given that partial or lack of information from the devices is easy to detect, these attacks are probably the easiest for prevention and for sending alarms. However, incomplete information


Over-infusion
SC config.: Unsafe programming of devices by coordinator (ox_ctrl, cap_ctrl, pump_ctrl input)
AC,BC,FC config.: Unsafe programming of pump through the coordinator (care_in)
OR
Programmed flow (Basal) rate is too high
Programmed drug concentration rate is too high
Programmed sampling rate of SPO2 and Cap rate is too low (delayed detection of over-infusion)
Programmed bolus flow-rate is too high
Programmed square bolus flow-rate is too high
Programmed square bolus duration is too long

(a)


Over-infusion
BC, FC: Alter EHR_data_in to sanity check caregiver infusion and bolus programming
FC: Alter EHR_data_in to sanity check coordinator's closed-loop control of the pump
Man-in-the-middle (MITM) attack between Coordinator and EHR to alter EHR_data_in

(b)

Figure 3: Attack Graphs representing attack vectors due to: (a) IMD initialization attacks, (b) EHR access attacks

at the coordinator may lead to incorrect decisions in emergency situations where data is time-sensitive.

- Incorrect Feedback Attacks: Represented in Figure 4 (b), these attacks represent the cases where the feedback received at the coordinator has incorrect information as a result of an adversary tampering with it. This can lead to wrong diagnosis, missed alarms and, in the FC configuration, incorrect actuation.
- Delayed Feedback Attacks: Represented in Figure 5, these attacks he cases where the feedback received at the coordinator is delayed as a result of an adversary. Such delayed feedback information may be interpreted as a current reading, causing untimely actuation and over-infusion.

In the feedback attack cases, we only show the avenues for attacks that can cause manipulation of the feedback to the coordinator. We do not attempt to describe the mechanisms for an attacker to perform such manipulation, since attempts to predict adversary behavior often lead to inadequate defenses. Instead, we focus on the broad outcomes of these attacks. Fortunately, many of the attacks can be thwarted with known countermeasures obtained from best practices in network security, software validation, and operating system security to ensure the attack cannot occur. However, one must be aware that attack vectors can be activated simultaneously by the attackers.

Broadly speaking all these attacks (with a trustworthy caregiver) are manifestations of the "confused deputy" attack [13]. In a confused deputy attack, a privileged entity (the "deputy") is manipulated by an attacker to perform an unsafe act. Depending on the attack scenario, the caregiver, the coordinator, and the pump can be victims of a confused deputy attack. While the exact details vary for each entity, the general pattern is the same: the attacker would block, alter, or delay the information the deputy requires for proper operation. This would cause the deputy to make a medical decision with inaccurate or limited information. As an example, we consider a confused deputy attack on the caregiver. If the attacker wants to manipulate the caregiver into over-infusing the patient with pain medication, the attacker may alter the sensor readings from the pulse oximeter and the capnograph. In particular, the attacker may alter both sensors to indicate the patient's respiration is normal or elevated, regardless of the patient's actual respiration behavior. Accordingly, the caregiver may believe it is safe to administer a greater quantity of medication than what the patient can handle. If the attacker continues to report healthy readings, despite suppressed respiration, the attacker may manipulate the caregiver into programming a larger dose of medication when it is unsafe to do so.

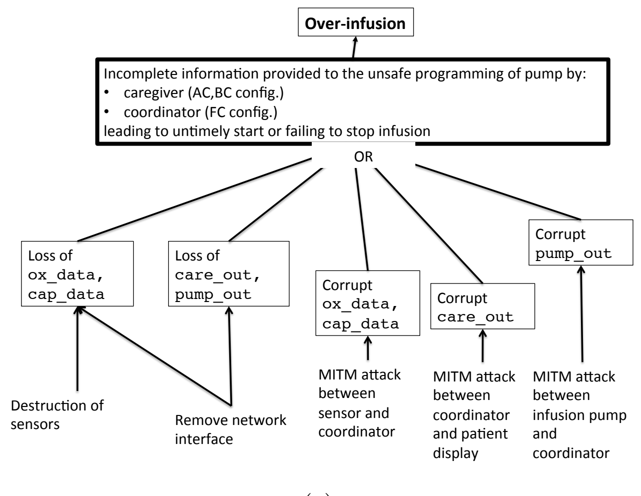
As the model changes to have greater coordinator involvement, the attack vectors shift. Once the coordinator has the responsibility of controlling boluses, an attacker can begin to manipulate the inputs to the coordinator with the goal of encouraging the coordinator to allow a bolus that it should prevent. In the FC configuration, the role of the caregiver is completely removed, placing these responsibilities in the coordinator. While the change in the FC configuration may seem to introduce a security risk, the attack vectors remain largely the same. The only difference is that the attacker must focus on manipulating the coordinator instead of the caregiver. The essential issue in the FC configuration is to design closed-loop control of the coordinator application to be safe from causing the patient harm.

4.2 Mitigating the Attacks

For over-infusion to occur, the infusion pump has to administer large quantities of pain medication in an untimely manner. The only method for an attacker to cause the controller to send the pump commands that trigger an over-infusion event are[1]:

- *Case 1 – Programming-Focused*: In this case, the care-

[1]Note that, in the above analysis we assume that the coordinator will not incorrectly program the settings to the PCA pump. The software on a coordinator must be carefully vetted and verified before it is used in a production environment to limit the chance of software errors.


Over-infusion
Incomplete information provided to the unsafe programming of pump by:
caregiver (AC,BC config.)
coordinator (FC config.)
leading to untimely start or failing to stop infusion
OR
Loss of ox_data, cap_data
Loss of care_out, pump_out
Corrupt ox_data, cap_data
Corrupt care_out
Corrupt pump_out
Destruction of sensors
Remove network interface
MITM attack between sensor and coordinator
MITM attack between coordinator and patient display
MITM attack between infusion pump and coordinator
(a)

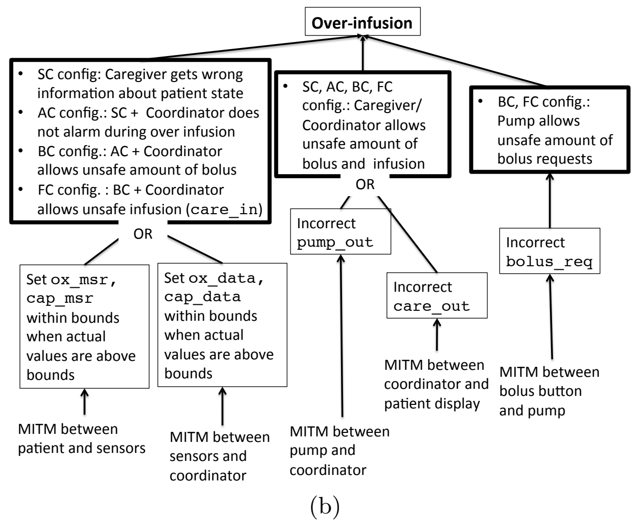

Over-infusion
SC config: Caregiver gets wrong information about patient state
AC config.: SC + Coordinator does not alarm during over infusion
BC config.: AC + Coordinator allows unsafe amount of bolus
FC config. : BC + Coordinator allows unsafe infusion (care_in)
SC, AC, BC, FC config.: Caregiver/ Coordinator allows unsafe amount of bolus and infusion
BC, FC config.: Pump allows unsafe amount of bolus requests
OR
OR
Set ox_msr, cap_msr within bounds when actual values are above bounds
Set ox_data, cap_data within bounds when actual values are above bounds
Incorrect pump_out
Incorrect care_out
Incorrect bolus_req
MITM between patient and sensors
MITM between sensors and coordinator
MITM between pump and coordinator
MITM between coordinator and patient display
MITM between bolus button and pump
(b)

Figure 4: Attack Graphs representing: (a) partial feedback attacks, (b) incorrect feedback attacks

giver's input to the device in the SC configuration is incorrect. The caregiver is not in our TCB and therefore can provide incorrect input to the devices (for SC) or the coordinator (AC and BC) either simply due to human error or incompetence. The caregiver can press incorrect keys when entering values, calculate rates incorrectly, or simply program the pump accurately, but use an incorrect concentration of the medication. These cases can be remediated using local solutions at the pump itself such as drug libraries, flow sensors, and barcode scanners [17]. One can push the remediation to the coordinator as well, but that would significantly increase the complexity of the coordinator, which is undesirable.

- *Case 2 – Communication-Focused*: The coordinator's input to the pump, `pump_in`, for the AC, BC and FC configurations, is incorrect. This is possible because: (a) some or all the information going out of the coordinator to the pump over `pump_in` has been altered (delayed, modified, corrupted) by adversaries, (b) bolus information going from the bolus button to the infusion pump over `bolus_req` has been altered (delayed, modified, corrupted) by adversaries; (c) some or all the information going into the coordinator from the sensors (i.e., `ox_data`, and `cap_data`), EHR (i.e., `EHR_data_in`), and the pump (i.e., `pump_out`) has been altered (delayed, modified, corrupted) by adversaries; and (d) the programming instructions from the caregiver to the coordinator, `care_in` has been altered (delayed, modified, corrupted). These can be prevented by using cryptographic primitives to preserve the confidentiality, integrity and authenticity properties of the lines of communication. Such techniques are considered best practices for securing network communication.

- *Case 3 – Hybrid* : The caregiver's programming of the coordinator, `care_in`, in the AC and BC and FC configurations, is incorrect. All the reasons listed above for Cases 1 and 2 may apply and the same prevention strategies can be used.

- *Case 4 – Behavior-Focused*: The pump itself is malicious and does not adhere to the coordinator or caregiver programming. This can happen if the pump has been physically altered by adversaries. In this case, attack prevention (as in the three aforementioned cases) becomes very difficult. The only option is to detect problems with the patient's health based on data from the sensors and raise an alarm. However, if the sensors are compromised as well, the system simply lacks sufficient data to raise an alarm. The only way to deal with this situation is through redundancy of sensors, assuming at least some of them are not compromised. Similarly, if the caregiver is malicious, then any programming of the system done by the caregiver may introduce an attack into the IMD system as well. This cannot be prevented unless there is redundancy in the patient treatment related decision-making.

In summary, these vectors characterize the varied types of misinformation that could reach the PCA pump, the coordinator, and the caregiver. Within each vector, the attacker can devise a variety of actual attacks. The context of the IMD deployment plays a big role in identifying them. Any mitigation solution for these attacks have to therefore consider all of these cases.

5. LESSONS LEARNED

The attack vectors in Figures 3, 4, and 5 highlight several important points:

- **Individual medical device safety does not equate to interoperability safety.** A device can be formally defined as "safe" if and only if none of its execution paths invoke a particular set of negative actions. However, the safety of a particular medical device and the coordinator are insufficient to ensure that it remains safe in an interoperable setting. In our system model, adversary induced misinformation or bad input can cause an infusion pump to over-infuse medication, endangering patient safety. This condition can occur even if the infusion pump is guaranteed to meet its own safety requirements as in work by Kim *et al.* [17].

- **Interoperability tries to replicate human processes.** The transition from SC all the way to FC simply represents the addition of sub-graphs in the attack graph. These sub-trees can only lead to over-infusion if the coordinator receives bad data or has faulty software or application. While the latter can be addressed with proper design and software verification techniques, the former condition is a simple transformation from today's caregiver scenario: rather than a human receiving inaccurate data, the coordinator receives it. The action taken is largely the same. Hence, it is not sufficient to develop safe coordinator unless it also has secure communication.

- **All security attacks are manifest as a confused deputy attack.** We assume that the pump software itself is designed to meet certain safety goals. Thus, the pump can only violate patient safety goals if it receives invalid input from a caregiver or coordinator. Likewise, when the coordinator and the caregiver are both considered trusted, patients can only be harmed if the pump is mis-programmed based on inaccurate/delayed/partial inputs from the sensors and EHR.

- **Best safety practices may thwart some attacks.** The techniques used to prevent data entry errors for caregivers, such as drug libraries, barcode scanners, and flow sensors, also play a role in preventing security failures. However, these techniques may not be exhaustive nor sufficient to thwart all security attacks. In particular, each of these devices and their interconnects must be trustworthy; otherwise, an attacker can simply tamper with the information they provide to the coordinator and pump.

- **Only pervasive misinformation attacks can silence the interoperability coordinator.** The sensor inputs `ox_data` and `cap_data`, plus the pump output `pump_out` and possibly the EHR, must simultaneously be manipulated; otherwise, an alarm may be raised. Such an attack would require manipulation between the coordinator and pump, along with incorrect sensor data, to be effective.

- **Attacks from compromised entities in the interoperability are difficult to prevent.** If any of the three main types of entities in the interoperability setup, namely the sensors, the caregiver, and pump can be compromised, then the traditional information security solutions described for securing the inputs are rendered moot. One can use redundancy to attempt to detect events of compromise, but this requires at least one uncompromised IMD.

- **Security may be the proper subset of safety for IMDs**. When privacy is not considered (as is the case in our analysis), security may be a subset of safety. If we do consider privacy, then loss of privacy may not always lead to immediate safety problems for the patient. We do note that reconnaissance and eavesdropping are often precursors to more active attacks and that privacy may itself be an important security and safety goal.

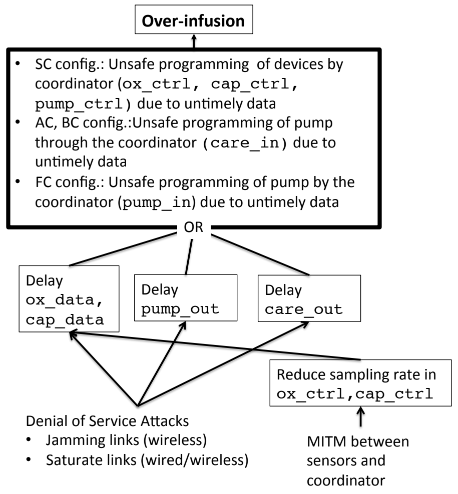

Over-infusion
• SC config.: Unsafe programming of devices by coordinator (ox_ctrl, cap_ctrl, pump_ctrl) due to untimely data
• AC, BC config.:Unsafe programming of pump through the coordinator (care_in) due to untimely data
• FC config.: Unsafe programming of pump by the coordinator (pump_in) due to untimely data
OR
Delay ox_data, cap_data
Delay pump_out
Delay care_out
Reduce sampling rate in ox_ctrl,cap_ctrl
Denial of Service Attacks
• Jamming links (wireless)
• Saturate links (wired/wireless)
MITM between sensors and coordinator

Figure 5: Attack Graphs representing delayed feedback attacks

6. RELATED WORK

Though some work has been done in developing frameworks for enabling interoperability between medical devices, little work has been done in exploring security issues for interoperable medical devices. King *et al.* [18] present an open-source Medical Device Coordination Framework (MDCF) for exploring solutions related to designing, implementing, verifying, and certifying systems of integrated medical devices. The goal of this project is to describe some of the details of enabling plug-n-play interoperability between medical devices that the ICE standard proposes. The framework supports a publish-subscribe architecture and uses a model-based programming environment for rapid development of IMD systems. The scope of this project has largely been on enabling interoperability and doing it safely in a certifiable manner [14]. Though security is nominally part of this framework, it has not been explored yet.

In our previous work [24], the security of ICE architecture was examined assuming the devices were using a wireless channel to communicate. The analysis was a very high level and was not specific to any interoperability setting. In later work [23, 25], we developed high-level models for classifying the security attacks and their consequences on interoperable medical devices. These models again did not deal in the specifics of a particular interoperability setup and consequently cannot be used to aid in designing security-conscious interoperability architectures. That being said, models developed from these efforts are certainly complimentary to this effort and can be incorporated to extend this work.

7. CONCLUSIONS AND FUTURE WORK

Medical device interoperability is an increasingly prevalent example of how computing and information technology will revolutionize and streamline medical care. However, one aspect that has not been considered thus far is ensuring IMDs do not inadvertently harm patients in the presence of malicious adversaries. This work outlines our effort in understanding the threats faced by IMDs. It is an important first step in eventually designing secure interoperability architectures. In this regard, we presented a detailed attack-graph-based analysis of threats on PCA interoperability under various levels of interoperability. Assuming a trusted coordinator, most of the attacks were discovered to be various forms of the confused deputy attack. We then described mitigation approaches possible for each of the possible attack classes. Many of the communication channel-oriented attacks can be mitigated using existing best-practices and available cryptographic solutions. However, behavioral attacks based on physical attacks on the devices themselves are very difficult to protect against technologically. Our analysis shows that individual medical device safety does not equate to IMD safety despite having a trusted coordinator.

In the future, we plan to extend the analysis to remove the entire coordinator from the trusted computing base and analyze the potential for attacks on constituents of the coordinator, namely the supervisor and network controller, the logs and the alarm system. We also plan to expand on this effort to design an interoperability architecture and coordinator that can handle many of the security problems that the coordinator in the ICE architecture cannot handle. Overall, we want to understand the relationship between safety and security in IMDs and other such medical cyber-physical systems (MCPS), which, as of now, is not entirely clear.

8. REFERENCES

[1] D. Arney, S. Fischmeister, J. Goldman, I. Lee, and R. Trausmuth. Plug-and-Play for Medical Devices: Experiences from a Case Study. *Biomedical Instrument & Technology*, 43(4):313–317, July 2009.

[2] ASTM F29.21. Medical devices and medical systems — essential safety requirements for equipment comprising the patient-centric integrated clinical environment (ICE) — part 1: General requirements and conceptual model.

[3] T. Choen. Medical and information technologies converg. *IEEE Eng. Med. Biol. Mag*, 23(3):59–65, May–June 2004.

[4] M. Clarke, D. Bogia, K. Hassing, L. Steubesand, T. Chan, and D. Ayyagari. Developing a standard for personal health devices based on 11073. In *EMBS*, 2007.

[5] T. Denning, K. Fu, and T. Kohno. Absence makes the heart grow fonder: New directions for implantable medical device security. In *HotSec*, 2008.

[6] T. Denning, Y. Matsuoka, and T. Kohno. Neurosecurity: security and privacy for neural devices. *Neurosurgical Focus*, 27(1), 2009.

[7] D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.

[8] K. Grifantini. Plug and Play Hospitals. `http://www.technologyreview.com/biomedicine/21052/`, July 2008.

[9] S. L. Grimes. Security: A new clinical engineering paradigm. *IEEE Eng. Med. Biol. Mag*, 23(4):80–82, July–August 2004.

[10] P. P. Gunn, A. M. Fremont, M. Bottrell, L. R. Shugarman, J. Galegher, and T. Bikson. The health insurance portability and accountability act privacy rule: A practical guide for researchers. *Med. Care*, 42(4):321–327, April 2004.

[11] D. Halperin, T. Heydt-Benjamin, B. Ransford, S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. Maisel. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *IEEE Security and Privacy*, 2008.

[12] S. Hanna, R. Rolles, A. Molina-Markham, P. Poosankam, K. Fu, and D. Song. Take two software updates and see me in the morning: The case for software security evaluations of medical devices. In *USENIX conference on Health security and privacy*, 2011.

[13] N. Hardy. The confused deputy: (or why capabilities might have been invented). *ACM SIGOPS Operating Systems Review*, 22(4):36–38, 1988.

[14] J. Hatcliff, A. King, I. Lee, A. Macdonald, A. Fernando, M. Robkin, E. Vasserman, S. Weininger, and J. Goldman. Rationale and architecture principles for medical application platforms. In *IEEE/ACM Third International Conference on Cyber-Physical Systems (ICCPS)*, pages 3–12, 2012.

[15] Health level seven international. `http://www.hl7.org/`.

[16] Integrating the healthcare enterprise. `http://www.ihe.net/`.

[17] B. G. Kim, A. Ayoub, O. Sokolsky, I. Lee, P. Jones, Y. Zhang, and R. Jetley. Safety-assured development of the gpca infusion pump software. In *Embedded Software (EMSOFT), 2011 Proceedings of the International Conference on*, pages 155–164, 2011.

[18] A. King, S. Procter, D. Andresen, J. Hatcliff, S. Warren, W. Spees, R. Jetley, P. Jones, and S. Weininger. An open test bed for medical device integration and coordination. In *Software Engineering - Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on*, pages 141–151, 2009.

[19] Michael Wong. Physician-patient alliance for health and safety improving health and safety through innovation and awareness how often do errors with patient-controlled analgesia (PCA) occur? `http://ppahs.org/2011/10/31/how-often-do-errors-with-pca-occur/`.

[20] E. Morris, L. Levine, C. Meyers, D. Plakosh, and P. Place. Systems of systems interoperability. Technical report, Carnegie-Mellon University, April 2004.

[21] N. L. Snee and K. A. McCormick. The case for integrating public health informatics networks. *IEEE Eng. Med. Biol. Mag*, 23(1):81–88, January–February 2004.

[22] A. Tolk, S. Diallo, and C. Turnitsa. Applying the levels of conceptual interoperability model in support of integratability, interoperability, and composability for system-of-systems engineering. *Journal of Systemics, Cybernetics and Informatics*, 5(5):65–74, 2007.

[23] E. Vasserman, K. Venkatasubramanian, O. Sokolsky, and I. Lee. Security and interoperable-medical-device systems, part 2: Failures, consequences, and classification. *IEEE Security & Privacy*, 10(6):70–73, 2012.

[24] K. Venkatasubramanian, S. Gupta, R. Jetley, and P. Jones. Interoperable medical devices. *Pulse, IEEE*, 1(2):16 –27, September–October 2010.

[25] K. Venkatasubramanian, E. Vasserman, O. Sokolsky, and I. Lee. Security and interoperable-medical-device systems, part 1. *IEEE Security & Privacy*,

10(5):61–63, 2012.