Toward a Safe and Secure Medical Internet of Things

Hamed Soroush, Ph.D
Senior Research Security Engineer
Real-Time Innovations
hamed@rti.com

David Arney
Lead Engineer, MD PnP Interoperability Program
Massachusetts General Hospital

Julian Goldman, MD
Medical Director of Biomedical Engineering for Partners HealthCare System
Director, MD PnP Interoperability Program
Massachusetts General Hospital
JMGOLDMAN@mgh.harvard.edu

1. INTRODUCTION

The landscape of modern medicine is dramatically changing with the advent of networked medical devices. This change brings the promise and the challenge of next-generation integrated medical systems that will interoperate efficiently, safely and securely. It is anticipated that it will significantly lower the rates of preventable medical errors, now estimated to be as high as the third leading cause of death in the U.S. [1]; and by providing improved patient outcome at lower costs [2]. Such improvements include, but are not limited to, support for real-time clinical decision support and automatic diagnosis, real-time checking of adverse reactions to medications, reduced false alarms and physiologic closed-loop control systems [5][6].

The grand vision of the Medical Internet of Things (MIoT) is to enable the deployment of patient-centric and context-aware networked medical systems in all care environments, ranging from homes and general hospital floors to operating rooms and intensive care units. Heterogeneous devices in each care environment would effectively share data – efficiently, safely and securely to minimize preventable errors that are often induced unknowingly by human operators. As medical devices move between different care environments or from patient to patient, they would securely discover other devices that they need to interoperate with, and then verify and execute safe, authorized and compliant operational profiles. The key to realizing this vision is coming up with standardized architectures that balance utility, reliability and safety requirements with those of security and privacy, and providing this information as a roadmap.

The Integrated Clinical Environment (ICE) framework, as defined by the ASTM F2761-09 standard [1], is a significant step toward enabling this interoperable MIoT vision. Most recently, with support from the US Government, we have been making advances to integrate security into ICE. Security considerations for interconnected and dynamically composable medical systems are critical not only because laws such as the Health Insurance Portability and Accountability Act (HIPAA) [4] mandate it, but also because security attacks can have serious safety consequences for patients. As these medical devices will be brought together and mixed/matched in an ad hoc fashion to serve the needs of a given patient (dynamically composed systems), additional security mechanisms will be required. They will need to support automatic verification that the system components are being used as intended in the clinical context, that the components are authentic and authorized for use in that environment, that they have been approved by the hospital's biomedical engineering staff and that they meet regulatory safety and effectiveness requirements.

As far as medical device communications is concerned, few of the existing or proposed standards for dynamically composed and interoperable medical devices and information systems include sufficiently comprehensive or flexible security mechanisms to meet current and future safety needs. There are significant gaps between required security properties and those that can be fulfilled even by combinations of currently standardized protocols [2]. Safety considerations in

these standardization efforts are effectively incomplete due to a lack of appropriate security analysis.

Regulators are also noting the importance of incorporating security for safety and privacy in the medical domain. The FDA is calling for medical device manufacturers to address cyber-security issues for the *entire* lifecycle of the device: from the initial design phase through deployment and end-of-life [8][9]. Although these calls are in the form of draft guidelines for ensuring device security and interoperability, there is evidence that the FDA intends to use them as a basis for clearing medical device submissions [26]. This seems to be addressing the traditional lack of incentive for medical device manufacturers to incorporate necessary security mechanisms in their products for fear of complicating regulatory approval [27].

In this paper, we present recent research on protecting the communications within ICE based on the fine-grained security mechanisms provided by the OMG Data Distribution Service (DDS) standard. In Section 2, we provide a background on ICE and the components that comprise ICE systems. We provide an overview of the DDS standard suite, which forms the connectivity platform of OpenICE [4]; OpenICE is the ICE reference implementation. We also briefly introduce the DDS Security architecture for granularly protecting DDS-based communications. Sections 3 and 4 go over our analysis, developed prototypes and results.

Real-Time Innovations (RTI) and the Medial Device Plug-and-Play (MD PnP) Program at the Massachusetts General Hospital have collaborated on this research. We are planning on applying our findings to the Industrial Internet Consortium's Connected Care Testbed.

2. BACKGROUND

2.1 Background on Integrated Clinical Environments (ICE)

The ICE framework, as defined by the ASTM F2761-09 standard [1] provides an approach for integrating heterogeneous medical devices and coordinating their activities to automate clinical workflows. From a high-level perspective, the idea behind ICE is to allow medical devices that conform to the ICE standard, either natively or using an after-market adapter, to interoperate with other ICE-compliant devices regardless of manufacturer. A similar paradigm has existed for many years in the personal computing domain, leading to an explosion of devices supporting WiFi, USB or Bluetooth standards. A similar approach in the medical domain, if done correctly, would enable dramatic improvements to patient safety. Known examples include patient transfers from the Operating Room (OR) to Intensive Care Units (ICU) or reducing false alarms in Patient-Controlled Analgesia (PCA) systems. In both of these examples, cross-vendor inter-device communications significantly reduces preventable medical errors [2].

Figure 1 depicts the general architecture of ICE and how it maps to the equipment of a test-bed setup at the MD PnP Interoperability Lab.

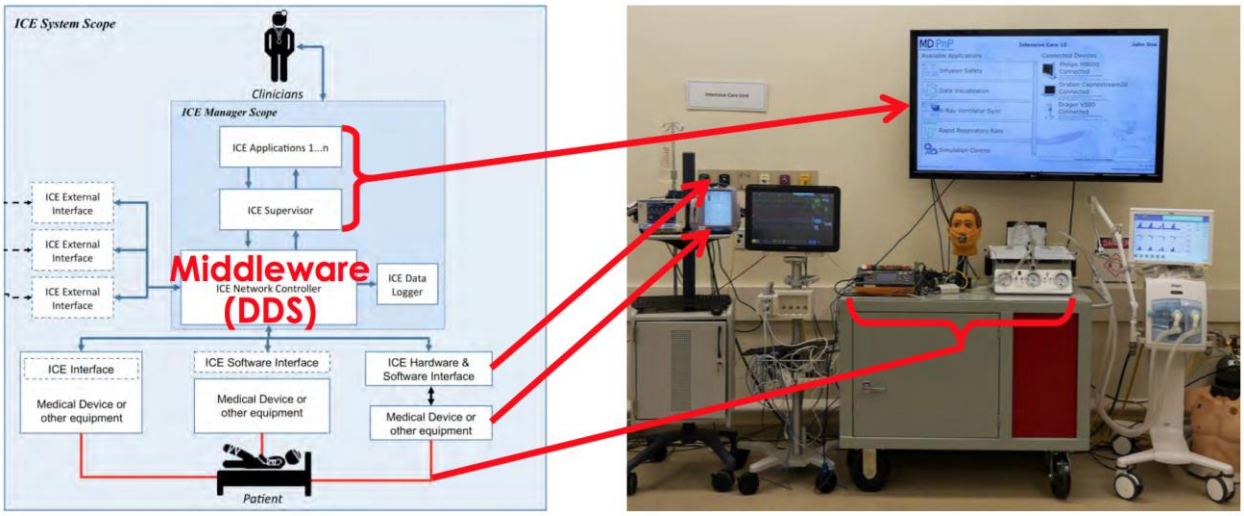

Figure 1. General architecture of ICE and an instantiation of it in a test setup at MD PnP Lab.

The *ICE Network Controller* is essentially a high-assurance middleware that forwards data or commands to or from ICE applications and devices, ensures communication quality-of-service and is agnostic as to the intended use of the clinical apps that it supports. It also manages the discovery and connection protocol for devices that wish to connect to the system. Given its critical communicatory role in ICE, having high-performance and context-aware security support in the network controller is paramount. The major functional security requirements for the network controller include: i) having authentication mechanisms for validating the identity of devices and apps, vouching for their provenance and ICE compliance, ii) having flexible yet easy-to-use mechanisms for defining and enforcing access control policies for various ICE configurations in different care environments, iii) having a mechanism for secure device and app discovery, iv) having a secure auditing mechanism and v) having mechanisms to guarantee the integrity, freshness and confidentiality of data. Note that the functional requirement should be met via a solution that has minimal negative impact on non-functional requirements such as performance, availability, robustness, and ease of use for clinicians and developers.

The *ICE Supervisor* provides separation/isolation-kernel-like data partitioning and time partitioning. It makes sure the information cannot inadvertently leak between apps and apps cannot inadvertently interfere with one another. It provides real-time scheduling guarantees that the computation in one app cannot cause the performance of another to degrade or fail. It also provides a console that allows a clinician to launch apps, monitor their progress and provide user-input during app execution. The ICE Network Controller and Supervisor may be incorporated together and deployed as a standalone ICE Manager.

ICE Applications are programs that accomplish a clinical objective by interacting with one or more devices attached to the network controller. As each app executes in the supervisor, it defines the intended use of the current ICE configuration. An important safety- and security-related concept

is that ICE medical devices never interact directly with each other; all interaction is coordinated and controlled via the ICE apps. It is crucial that ICE apps exactly correspond to the specified task they were designed for.

The *ICE Data Logger* is dedicated to logging communication and other important events within the Network Controller and Supervisor. The data logger should also record security-related events.

ICE Equipment Interfaces declare the functional capabilities of the device (e.g., format of its data streams, commands to which it responds) along with non-functional properties of the data such as the rate at which data elements are streamed from the device. It is crucial that ICE Interfaces are designed with considerations for usable security, for both developers and clinical end-users.

2.2 Background on Data Distribution Service: A Communication Platform for ICE

Many communication standards have been proposed for dynamically composable and interoperable medical devices and information systems. Unfortunately, few of them include security mechanisms that are flexible and comprehensive enough to meet current and future safety needs [2]. In fact, recent work [2] has shown that there are significant gaps between required security properties for these systems and those that can be addressed even by a combination of currently standardized protocols. Safety considerations in these standardization efforts are effectively incomplete due to a lack of appropriate security analysis. Unfortunately, the promising ICE standard is no different. To address this, we developed a prototype of ICE based on RTI's implementation of the Object Management Group (OMG) Data Distribution Service (DDS) [3] as the ICE Network Controller, with the hopes of identifying & addressing a number of such gaps.

DDS is a communications API and an interoperability standard that provides a data-centric publish-subscribe model for integrating loosely coupled real-time distributed systems. A key feature of DDS is that it is *data-centric* in the sense that it separates state management and data distribution from application logic and supports discoverable data models. This exposes the data model to the communication middleware, enabling the DDS middleware to reason about and optimize the performance of data movement within the system. In order to customize run-time behavior and achieve a desired performance profile, DDS allows publishing and subscribing entities to express several quality-of-service (QoS) parameters. The offered versus requested QoS requirements of the participating entities are matched before any communication can proceed. The standard DDS QoS parameters include durability, reliability, deadline, resource limits, ownership, liveliness and several others [3].

DDS is currently being used as an Industrial Internet connectivity platform in many critical applications [10] within healthcare [5][11][12] [13][14][15][16][17][24][25], energy [21], transportation [20], and defense [22] sectors.

2.3 Data Distribution Service Security

The OMG DDS Security Specification adds support for authentication, authorization, access control, confidentiality, integrity and non-repudiation for the data sent over DDS. Moreover, it provides a security auditing capability to evaluate the overall communication state. Due to the data centric design of DDS, DDS Security can provide *fine-grained access control* over the messages and sub-messages that include both data and meta-data. This allows DDS to control and enforce which applications have authorization to publish and subscribe to the numerous data types on the network.

DDS Security is designed to handle scalable deployment scenarios, specifically the one-to-many (multicast) distribution of encrypted information while maintaining real-time quality-of-service. It also provides an extensible plugin-based architecture, as well as a set of built-in plugins for out-of-the-box interoperability. This architecture allows application developers to integrate with pre-existing identity management mechanisms, authorization policy repositories or cryptographic libraries, which might be program-specific.

Figure 2 shows the pluggable architecture of DDS Security. The authentication plugin supports identity verification, mutual authentication and shared secret establishment. The access control plugin enforces granular security policies. The cryptographic operations, such as encryption, decryption, hashing, digital signatures and key derivation are implemented in the cryptographic plugin. Finally, logging and data tagging plugins are used for auditing security-relevant events and annotating data with a security label, respectively.

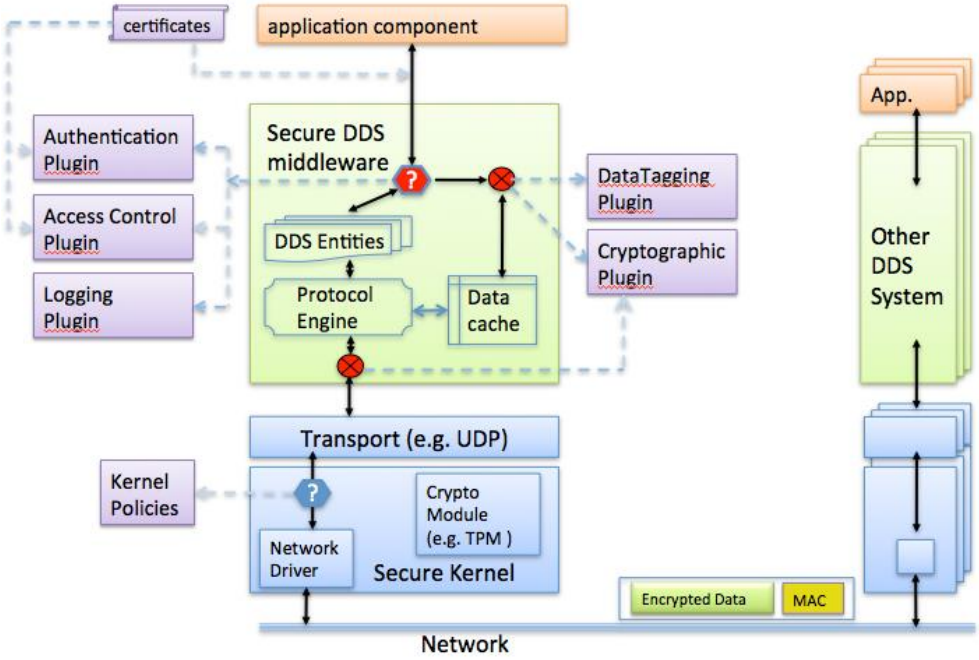

Figure 2: Architectural View of DDS Security

3. PRIMARY ANALYSIS

Our foremost objective towards laying down the foundation to secure clinical environments was to identify security risks, threats and requirements of various clinical scenarios. These are listed in the table below. Our findings have been mostly consistent with some of the existing literature on the topic [6][7] as far as external attackers are considered. However, we found that an important yet often neglected requirement is to minimize the impact of insider attacks posed by already-compromised devices that are unknowingly used in ICE settings. We discuss such an attack to instantiations of ICE that utilize secure transports such as TLS in Section 4.

Attack Class	Description	Susceptible Components
Destroy	Physically destroy ICE components; e.g. cut an infusion pump tube.	All Architectural Components of ICE
Disturb	Modify exchanged data to prevent correct operation of components; e.g. man-in-the-middle or replay attacks	All Architectural Components of ICE
Reprogram	Modify data or code in an ICE component to prevent its correct operation; e.g. modify infusion pump software to deliver extra medication	All Architectural Components of ICE Except the Communication Network Itself
Denial of Service	Exploit bugs or interfaces that were not designed with security in mind	All Architectural Components of ICE
Eavesdrop	Listen in on the deployed ICE environment to learn sensitive information.	Communication Network

Table 1. General attack model for ICE as identified in [6]

Use of an ICE controller based on DDS Security potentially addresses or mitigates Disturb, Denial of Service and Eavesdrop attacks. Further, it would mitigate the impact of insider attacks dramatically.

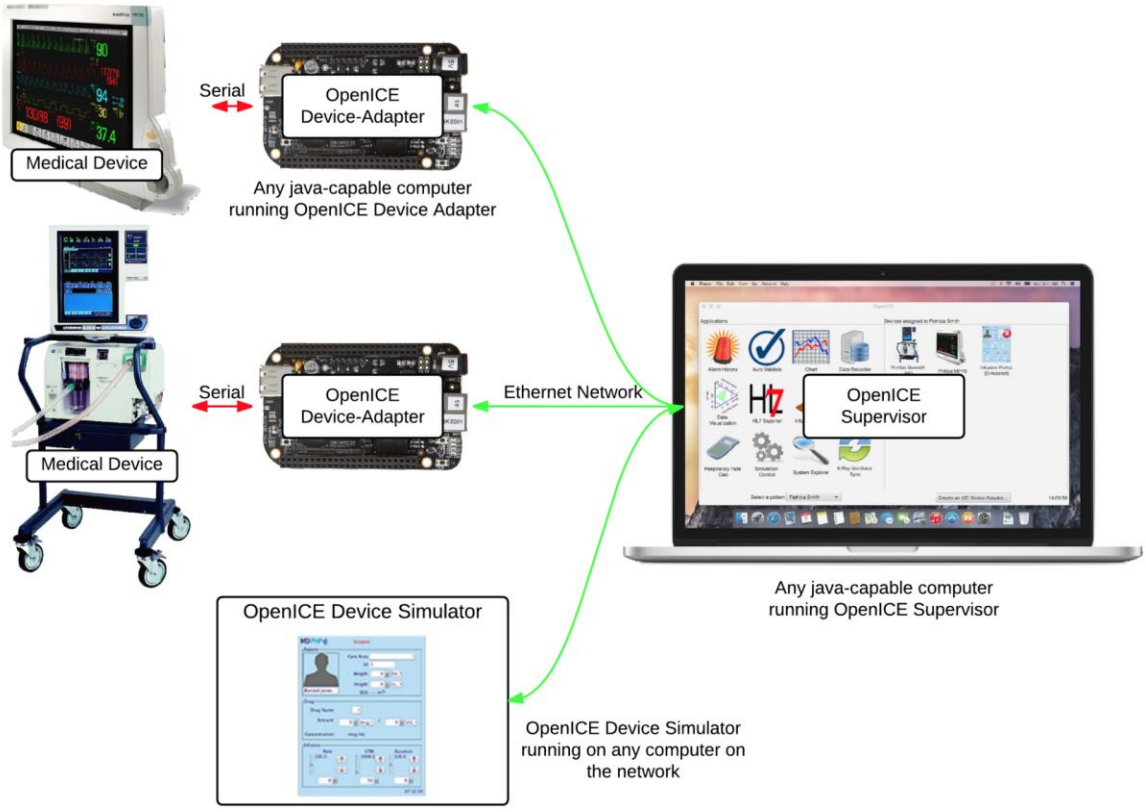

Figure 3. OpenICE – Developed by MD PnP Lab, it enables connectivity between various types of devices.

4. PROTOTYPE DEVELOPMENT

We designed and implemented two different prototypes, each supporting all of the medical applications (e.g. PCA Safety & Smart Alarms) provided in the OpenICE environment. OpenICE is an open-source reference implementation of ICE released by MD PnP lab. Figure 3 shows how OpenICE enables connectivity between various types of devices.

4.1 Practical Security Attacks on Current OpenICE Platform

Prior to the development of the prototypes, we verified that OpenICE, without any explicit security measure, can be easily attacked, endangering patient safety and privacy. We developed customized sniffers and injectors that an external attacker could use to eavesdrop on ICE communications or disturb device behavior (e.g. stop drug infusion, or inject wrong sensor readings).

4.2 First Prototype: OpenICE Using DDS on Top of Secure Transports

Our first prototype integrates OpenICE with RTI Connext DDS as the Network Controller, running on top of TLS or DTLS transports. In this prototype, security measures such as confidentiality or integrity of exchanged messages are not applied at the ICE Network Controller level, but at the

transport level that it uses. A fundamental research question here is whether such widely used, communication protocols provide acceptable security and performance for ICE.

While transport-level security provides typically reasonable protection against external attackers, it is not without limitations. Transport-level solutions do not provide any mechanism for granular access control. Even though these solutions protect the communication channel from external eavesdropping or packet injection, they do not provide any access control mechanism for data streams happening within the same protected link. Consequently, solutions based on them are vulnerable to insider attackers, as we demonstrate in our second prototype.

Transport-level security is also not sufficiently flexible to balance security versus performance. All messages that pass through the established secure link will be encrypted and authenticated, imposing an overhead that may not be necessary in many use cases. For example, risk analysis of an ICE system might conclude that encrypting temperature values from a sensor in a public room is not required and it is only needed to make sure sensor readings are authenticated. Being able to fine-tune security measures based on risk is especially important for resource-constrained devices or large-scale ICE or MIoT systems with bandwidth or delay sensitive applications. Further, such fine-tuning should ideally happen with minimal, if any, changes to the code base, as the code may not be available for modification or too costly to be modified.

Another issue with widely used transport-level security solutions such as TLS and DTLS is the lack of support for multicast. Multicast support has proven extremely useful for efficient and scalable discovery and information exchange in industrial systems.

4.3 Second Prototype: OpenICE Using RTI Connext DDS Secure

In the second prototype, we integrated OpenICE with RTI's implementation of the beta version of DDS Security Specification as the Network Controller. We also made sure that the integrated solution works with RTI Routing Service, acting as an intelligent gateway connecting multiple ICE environments. Such integration would ease adoption of ICE in fragmented hospital networks or in cases where ICE systems belong to different administrative domains.

RTI Routing Service is a software solution that provides the ability for unmodified new and legacy applications to interoperate, even if they were not originally designed to work together. It can be used to integrate different system or bridge to legacy messaging and networking technologies. It is used to form logical partitions for DDS systems across LANs or WANs or to bridge non-DDS systems provided that appropriate DDS adapters are linked to it [10]. Utilizing the Routing Service as an intelligent gateway enables a variety of security administration use cases in ICE. An example would be to segregate insecure legacy medical devices into separate administrative domains without disconnecting them from the secure ICE environment. This allows for a different, likely more strict, set of security policies to be applied to the legacy devices, while still keeping them connected to ICE.

We used DDS Security Built-in Plugins to protect ICE Network Controller operations. Table 2 shows capabilities of built-in plugins.

Authentication	X.509 Public Key Infrastructure (PKI) with a pre-configured shared Certificate Authority (CA) RSA or ECDSA Signature Algorithm for authentication, DH or ECDH for shared secret
Access Control	Configured by domain using a (shared) Governance file Specified via permissions file signed by shared CA Control over ability to join systems, read or write data topics
Cryptography	Protected key distribution AES128-GCM and AES256-GCM for authenticated encryption AES128-GMAC or AES256-GMAC for message authentication and integrity
Data Tagging	Tags specify security metadata, such as classification level Can be used to determine access privileges (via plugin)
Logging	Log security events to a file or distribute securely over DDS

Table 2: Capabilities of DDS Security built-in Plugins

The current commercial DDS Security Plugins rely on an existing public-key infrastructure (PKI) to be in place. Management of the PKI is outside the scope of DDS Security and industry best practices can be used. For our prototypes, we used a self-signed certificate authority.

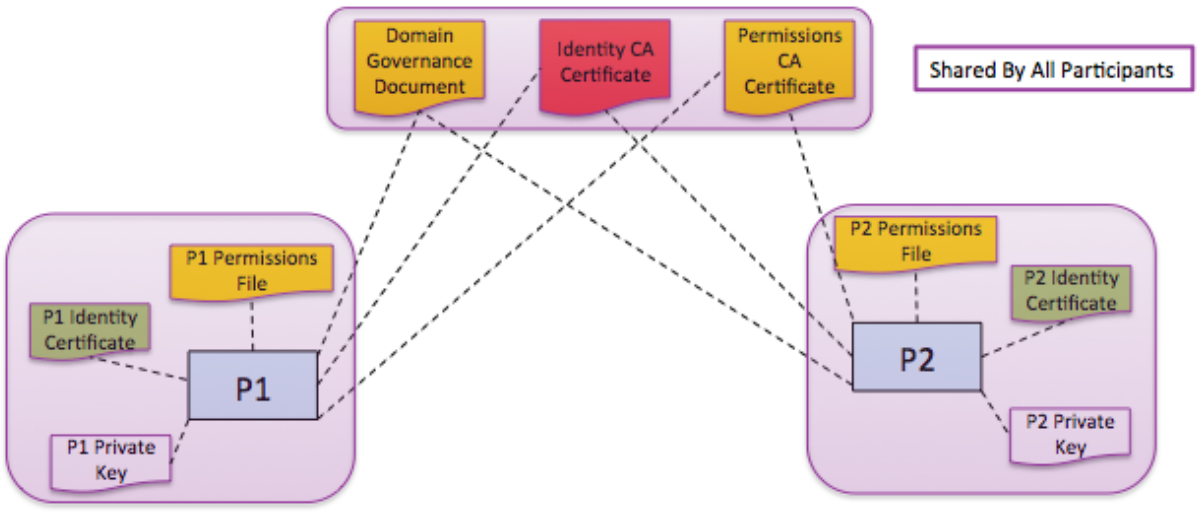

Figure 4: Deployment and configuration of DDS Security for two participants.

To operate using RTI's built-in security plugins, each DDS Domain Participant[1] requires 1) a public/private key pair, with the public key signed by a trusted certificate authority (referred to as the identity CA) forming an identity certificate, 2) permissions file signed by a trusted certificate authority (referred to as the permissions CA), 3) a DDS domain governance file, signed by the permissions CA, 4) an Identity certificate of the Identity CA, and 5) an Identity certificate of the Permissions CA. Figure 4 shows a possible deployment of DDS Security with two participants, P1 and P2. In an ICE setting, they could be an Oximeter and an ICE supervisor respectively.

The domain governance document is written in XML (eXtensible Markup Language), and specifies which DDS domains shall be protected, along with the details of the protection. The domain governance document is signed by the permissions CA and configures the following security aspects of the DDS domain: whether the discovery information should be protected and the kind of protection (MAC or ENCRYPT_THEN_MAC); whether liveliness messages should be protected; whether a discovered participant that cannot authenticate or fails authentication should be allowed to join the domain and see any data configured as unprotected; whether metadata (e.g., sequence numbers, heartbeats) should be protected and how; whether the payload should be protected and how; and whether read/write access to the topics should be open to all or restricted to the participants with proper permissions.

The XML permissions document contains the permissions of the domain participant, including which DDS domains it can join, what topics it can read or write, and what tags are associated with it.

4.4 Security Attacks on ICE When Run on Secure Transports

We implemented effective attacks against ICE when the Network Controller uses secure transports such as TLS or DTLS. Our attacks are based on ICE Infusion Safety App, which utilizes closed loop control of medical devices for safe delivery of patient controlled analgesia (PCA). The application controls the administration of IV medication and is programmed to stop the pump if it detects that the patient is in a non-normal state. Patient state is inferred from the readings of devices such as oximeters and capnographs. Figure 7 demonstrates a simplified ICE infusion safety application scenario, with topics published to or subscribed by various ICE components (see OpenICE Infusion Safety App Architecture [26] for further details).

[1] A DDS *domain* is a concept used to bind individual applications together for communication. To communicate with each other, *DataWriters* and *DataReaders* must have the same *Topic* of the same data type and be members of the same *domain*. Applications in one domain cannot subscribe to data published in a different domain. DomainParticipant objects enable an application to exchange messages within domains. *DomainParticipants* are used to create and use *Topics, Publishers, DataWriters, Subscribers,* and *DataReaders* in the corresponding *domain*.

In our attack, a compromised pulse oximeter publishes Alarm Limits associated with an uncompromised capnograph, either masking an alarm when it should happen (e.g. in case of a drug overdose) or when it shouldn't (e.g. causing alarm fatigue). Even though all communication in this attack scenario is encrypted and authenticated, a compromised insider device can cause system-wide damage, simply because what it can or cannot publish is not enforceable. DDS Security allows for fine-grained access control per device, preventing this significant type of attack.

In the second prototype, each ICE device has a cryptographically signed permission file that specifically indicates what topics can be published or subscribed by it. In order to recreate the original attack on this new framework, the attacker would have to hack into the public-key infrastructure (PKI) used in the framework, which is considered a much more difficult task if PKI is managed properly. In any case, if the PKI infrastructure becomes compromised, any cryptographic approach based on it will fail, be it based on TLS/DTLS or DDS Security.

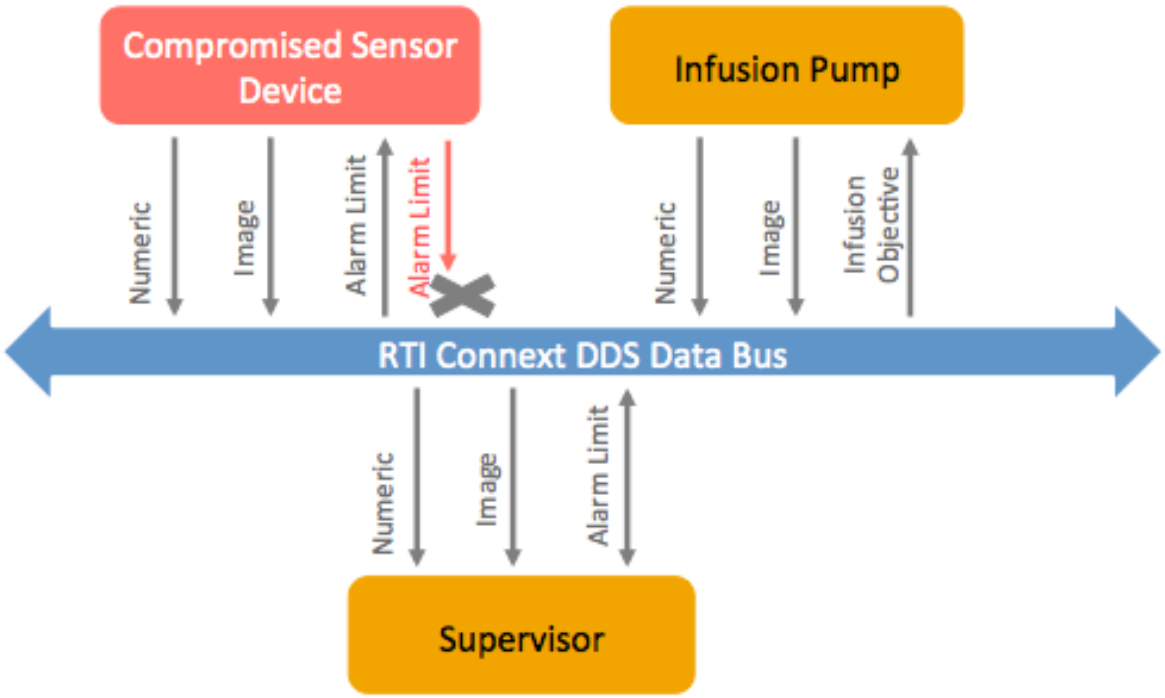

Figure 7. Simplified Architectural Diagram of OpenICE Infusion Safety App

Boxes represent ICE devices, and arrows represent topics that each device either publishes or subscribes to. The box in red represents a compromised oximeter that, in principle, should not be allowed to publish AlarmLimit topic data. AlarmLimit topic data should only be published by

the ICE supervisor and no other device, even if they are correctly authenticated. Both DDS Security and a secure transport such as TLS/DTLS allow for certificate-based authentication of devices, but use of DDS Security also enforces granular access control. Granular access control provides further resilience in presence of insider attackers, preventing system-wide damage such as the one discussed above.

5. CONCLUSION & FUTURE WORK

The grand vision of the Medical Internet of Things is to enable the deployment of patient-centric and context-aware networked medical systems in all care environments, ranging from homes and general hospital floors, to operating rooms and intensive care units. The key to realizing this vision is to come up with standardized architectures that balance utility, reliability and safety requirements with those of security and privacy. The ICE framework, as defined by the ASTM F2761-09 standard is definitely an important step toward enabling interoperable MIoT, however, it does not yet explicitly address security concerns.

In this paper, we presented recent research on protecting communications within IICE based on the fine-grained security mechanisms provided by the OMG DDS Security specification. We developed the two prototypes that respectively utilize secure transports (TLS/DTLS) and the DDS Security Architecture, and demonstrated why transport-level security solutions may not provide sufficient resilience against insider attacks utilizing authenticated but compromised medical devices.

In the future, we will work on defining and enforcing holistic security policies for ICE, integrate with endpoint protection mechanisms (e.g. secure Operating Systems, hardware-based root of trust), integrate with security management and monitoring solutions and explore issues at the intersection of usability and security in MIoT systems in general and ICE systems in particular.

6. REFERENCES

[1] ASTM F2761, 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, 2013.

[2] Foo Kune, D. a. (2012). Toward a Safe Integrated Clinical Environment: A Communication Security Perspective. Proceedings of the 2012 ACM Workshop on Medical Communication Systems (pp. 7--12). New York: ACM.

[3] OMG Data Distribution Service Standard: http://www.omg.org/spec/DDS/1.2/

[4] OpenICE: https://www.openice.info/

[5] RTI Customer Snapshot: DocBox: http://www.rti.com/docs/DocBox.pdf

[6] K. K. Venkatasubramanian et al. "Security and Interoperable-Medical- Device Systems, Part 1," IEEE Security Privacy, vol. 10, no. 5, pp. 61-63, Sept./Oct. 2012.

[7] Eugene Y. Vasserman , Krishna K. Venkatasubramanian , Oleg Sokolsky , Insup Lee, Security and Interoperable-Medical-Device Systems, Part 2: Failures, Consequences, and Classification, IEEE Security and Privacy, v.10 n.6, p.70-73, November 2012.

[8] Content of Premarket Submissions for Management of Cybersecurity in Medical Devices, Guidance for Industry & FDA Staff, October 2014

[9] Postmarket Management of Cybersecurity in Medical Devices, Draft Guidance for Industry and FDA Staff, January 2016

[10] RTI Customers: https://www.rti.com/industries/index.html

[11] ROS on DDS http://design.ros2.org/articles/ros_on_dds.html

[12] RTI Customer Snapshot: EMS Device Integration Platform for World's largest EMS equipment Provider. https://www.rti.com/industries/#HEALTH

[13] RTI Customer Snapshot: Minimally Invasive Robotic Surgery. http://www.rti.com/docs/German_Aerospace_Center_DLR.pdf

[14] RTI Customer Snapshot: Exelis C4i Command and Control Systems. https://www.rti.com/industries/#HEALTH

[15] RTI Press release: GE Healthcare. https://www.rti.com/company/news/ge-healthcare.html

[16] RTI Customer Snapshot: Medical Imaging. https://www.rti.com/industries/#HEALTH

[17] RTI Customer Snapshot: Advanced Proton Therapy. https://www.rti.com/industries/#HEALTH

[18] James, John T. PhD. A New, Evidence-based Estimate of Patient Harms Associated with Hospital Care. Journal of Patient Safety, September 2013. http://journals.lww.com/journalpatientsafety/Fulltext/2013/09000/A_New,_Evidence_based_Estimate_of_Patient_Harms.2.aspx

[19] Healthcare Technology. "Deaths by medical mistakes hit records" http://www.healthcareitnews.com/news/deaths-by-medical-mistakes-hit-records

[20] RTI IIoT Transportation Applications:. https://www.rti.com/industries/#TRANSPORT

[21] RTI IIoT Energy Applications: https://www.rti.com/industries/#ENERGY

[22] RTI IIoT Defense Applications: https://www.rti.com/industries/#DEFENSE

[23] Defense in Depth: A practical strategy for achieving Information Assurance in today's highly networked environments" https://www.nsa.gov/ia/_files/support/defenseindepth.pdf

[24] SmartAmerica Closed-Loop Healthcare Group: http://www.mdpnp.org/smartamerica.php

[25] OpenICE Infusion Safety App Architecture: https://www.openice.info/docs/3_apps.html - infusion-safety

[26] Mullen, A. B. (2013, 09). Premature enforcement of CDRH's draft cybersecurity guidance. http://www.fdalawblog.net/fda_law_blog_hyman_phelps/2013/09/premature-enforcement-of-cdrhs-draft-cybersecurity-guidance.html

[27] M. Rushanan, D. F. Kune, C. M. Swanson, and A. D. Rubin, "SoK: Security and Privacy in Implantable Medical Devices and Body Area Networks", IEEE Symposium on Security and Privacy, 2014

- Return to IIC Journal of Innovation 2nd Edition landing page for more articles
- Download the IIC Journal of Innovation 2nd Edition
- Visit the IIC Journal of Innovation 1st Edition landing page for more thought leadership