



WhiteScope

Security Evaluation of the Implantable Cardiac Device Ecosystem Architecture and Implementation Interdependencies

Billy Rios
Jonathan Butts, PhD

May 17, 2017

CONTENTS

Introduction	3
Architecture and Implementation Interdependencies	4
Related Research	6
Findings	7
<i>Obtainability of Vendor Subsystems from Public Sources</i>	7
<i>Commercial-off-the-shelf Microprocessors</i>	8
<i>Embedded Device Debugging Interfaces</i>	9
<i>Packed, Obfuscated or Encrypted Firmware</i>	11
<i>Use of ASCII Text Function Names and Software Debugging</i>	11
<i>Use of Third-Party Libraries</i>	13
<i>Mapping Firmware Image Into Protected Memory</i>	13
<i>External USB Connections</i>	14
<i>Hardcoded Credentials and Infrastructure Data</i>	14
<i>RF Activation</i>	15
<i>Remote Firmware Update</i>	16
<i>Digitally Signed Firmware</i>	16
<i>Removable Media/Hard-Drives</i>	16
<i>Encryption</i>	17
<i>Unencrypted Patient Data</i>	19
<i>Authentication to Conduct Programming</i>	20
<i>Physician Programming Applications</i>	20
<i>Dual Use of Radio Hardware for Home Monitoring Device and Physician Programmer</i>	20
<i>Command Whitelisting</i>	20
<i>Universal Authentication Token</i>	20
<i>Summary of Findings</i>	21
Evaluation of Security Controls	23
Conclusions	26
References	26

INTRODUCTION

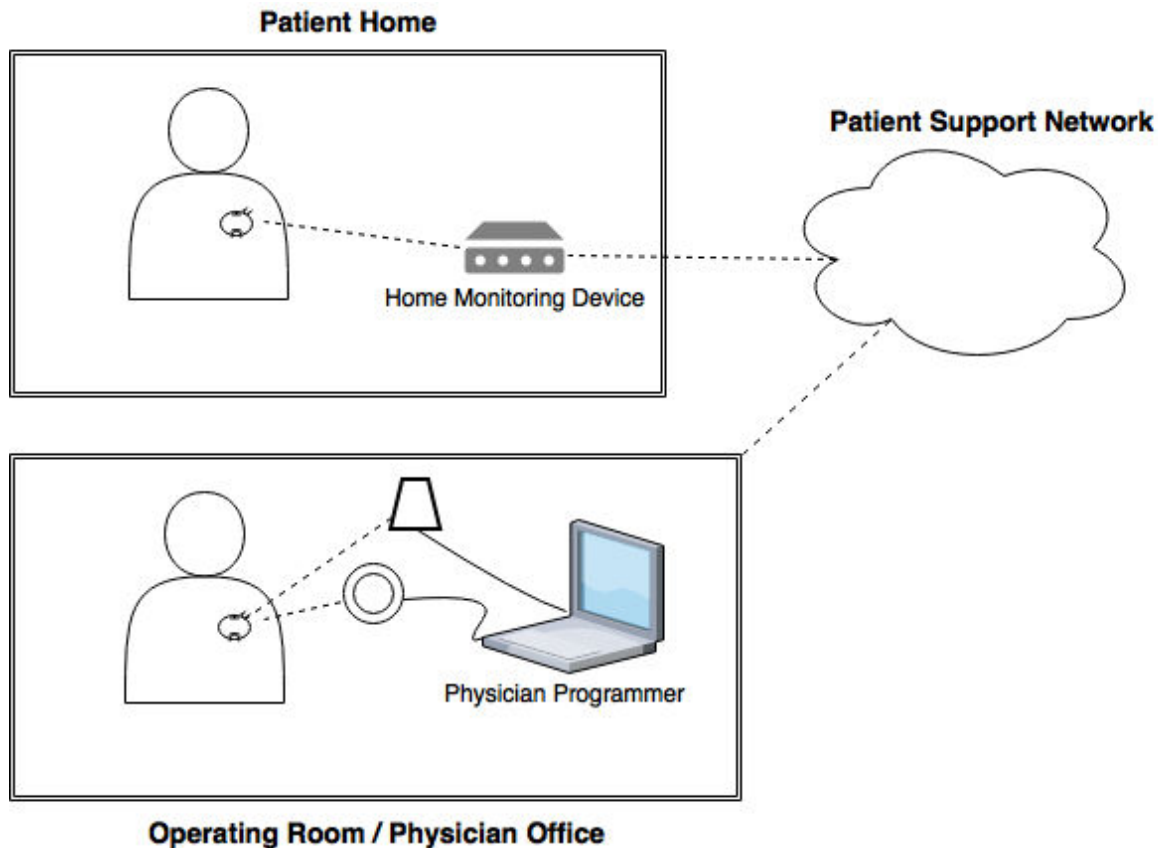
Based on recent security concerns and events, WhiteScope performed an exhaustive security evaluation on the implantable cardiac device ecosystem. This paper describes findings from the research, highlighting the principal security concerns associated with the implantable cardiac device ecosystem architecture and implementation interdependencies.

WhiteScope has obtained physician programmers, home monitoring devices, and implantable cardiac devices for the four major implantable cardiac device vendors. Conceptually, the four major vendors employ a similar architecture framework, including communication protocols, device intercommunications, embedded device hardware, and device authentication. Analysis revealed potential security risks stemming from the underlying protocols and system-to-system communications involving embedded devices. To mitigate potential impact to patient care, it is recommended that vendors evaluate their respective implementations and validate that effective security controls are in place to protect against identified deficiencies that may lead to potential system compromise. To aid in this process, WhiteScope provides questions that vendors can use to evaluate their respective implementations.

The results of the holistic analysis help clarify the nature and scope of the threats facing the implantable cardiac device ecosystem and the potential impact to patient care. WhiteScope researchers are motivated by the prospect of enhancing cyber security for the medical device community in a manner that strengthens patient safety.

ARCHITECTURE AND IMPLEMENTATION INTERDEPENDENCIES

The implantable cardiac device ecosystem consists of an implantable medical device (e.g., pacemaker or defibrillator), physician programmer, home monitoring device, and patient support network. The figure below shows a representation of the general ecosystem architecture.



The subsystems within the ecosystem interact to facilitate the delivery and monitoring of patient therapy. The following list identifies the primary functions of the ecosystem subsystems.

Implantable cardiac device: A battery-powered device that is surgically implanted under a patient's skin. A pacemaker implantable cardiac device sends electrical pulses to the heart to help the heart maintain a regular rhythm. The implantable cardioverter defibrillator (ICD) is the other primary type of implantable cardiac device. The ICD monitors heart rhythms and delivers an electrical shock if it senses dangerous rhythms.

Physician programmer: A device that is used in diagnosis and programming of the implantable cardiac device. The physician programmer is intended for clinical settings, such as the operating room and physician office. Once the cardiac device is

implanted, the physician programmer is used in the operating room to test the cardiac device functionality and set patient therapy parameters. Physicians may also use the programmer in follow-up office visits to examine cardiac device functionality, review administered therapy, and update patient therapy parameters. The physician programmer communicates with the implantable cardiac device via wireless communications using RF and inductive telemetry so that the device does not have to be surgically removed for updates and diagnosis.

Home monitoring device: A device that is used for transmission and monitoring of the implantable cardiac device and patient therapy data. The home monitoring device is intended for use at a patient's residence. The home monitoring device is often placed in proximity to the patient where they sleep. The home monitoring device gathers patient therapy data from the implantable cardiac device and transmits the data through the patient support network to the patient's physician. The incorporation of the home monitoring device into the ecosystem is intended to enhance patient care by identifying problems quickly and minimizing reoccurring office visits. Recently, the Heart Rhythm Society published research highlighting the clinical benefits of remote transmission and monitoring of implantable cardiac device and patient therapy data [6]. The findings indicate greater patient retention and improved adherence to scheduled evaluations, resulting in overall improvement to the quality and efficiency of patient care.

Patient support network: A dedicated networking infrastructure that is used to facilitate transmission of patient therapy data from the home monitoring device to the clinical physician. Associated communications media from the home monitoring device to the patient support network include dial-up modems, cellular and wifi. Vendors also use the patient support network to register patients and devices as well as to perform system updates to the home monitoring device. A portal associated with the patient support network provides patients and physicians the ability to login and review patient therapy data. The patient support network also provides capabilities to alert physicians when a patient exhibits certain cardiac parameters.

Communication within the ecosystem primarily occurs between: (i) the physician programmer and the implantable cardiac device for programming and analysis; (ii) the home monitoring device and the implantable cardiac device to query the implantable cardiac device; (iii) the implantable cardiac device and the home monitoring device to provide patient therapy data; (iv) the home monitoring device and the patient support network for relaying patient therapy data and to perform home monitoring device updates. The majority of communications rely on device-to-device protocols associated with embedded device communications.

RELATED RESEARCH

Other supporting research has been published that highlights security risks associated with the implantable cardiac device ecosystem. In 2008, Halperin *et al.* evaluated the security properties and presented attacks on common ICDs [2]. Their work demonstrated reverse engineering of ICD communications protocols and crafting attacks via a software defined radio that have the potential to impact patient safety. Similarly, Hei *et al.* presented research that demonstrated resource depletion attacks against IMDs [3]. The research focused on generic IMD implementations and claimed to provide the ability to significantly reduce the battery life for classes of IMDs that use wireless communications with an external programmer.

In 2010, Maisel and Kohno called for a specific regulatory framework for medical device security [4]. Their research emphasized concerns with security properties associated with devices that perform life saving functions such as a pacemaker. Additionally, Maisel and Kohno's work highlighted risks associated with architecture and implementation interdependencies for an ecosystem that extends beyond purely vendor implementation risks. They also argued that these medical devices provide important health benefits to patients and that security controls should be weighed accordingly against impact to patient care.

In 2012, Bursleson and Fu discussed design challenges associated with the security of implantable medical devices [1]. The authors provided a threat model and discussed how important security considerations apply commonly across many implantable cardiac devices. Indications of their research show common threads across various vendor IMDs, demonstrating broader domain-specific concerns.

Recently, Marin *et al.* published research that examined proprietary protocols used in wireless communications between physician programmers and ICDs [5]. They used black-box reverse engineering techniques to examine the long-range RF channel for the most recent generation of ICDs. Their research identified multiple protocol and implementation weaknesses that provide an attacker the potential to conduct privacy and denial-of-service attacks as well as initiate spoofing and replay attacks of messages that have the potential to impact patient safety. The authors note that their research was validated for at least ten types of ICDs.

The WhiteScope research presented in this document examines the architecture and implementation interdependences across the implantable cardiac device ecosystem. Our research focuses on a holistic analysis of the underlying architecture. The research findings, coupled with other supporting research, indicate potential security concerns with the core of the underlying architecture that are applicable across vendor implementations.

FINDINGS

As a whole, the implantable cardiac device ecosystem inherits security features associated with the underlying system-of-systems architecture. If adequate security controls are not implemented, weaknesses associated with architecture and implementation interdependencies have the potential to compromise ecosystem confidentiality, integrity, and/or availability - resulting in potentially negative consequences to patient care if those weaknesses are exploited.

WhiteScope initiated the analysis by examining architecture attributes and implementation interdependencies to identify potential risk areas. Once the risk areas were identified, WhiteScope obtained subsystems for the four major vendors and examined the subsystems to evaluate security controls and discern the existence of potential security weaknesses. The findings presented below show the results of the evaluation and identify potential security concerns that may warrant the implementation of further security controls to mitigate potential risks. The findings do not attempt to capture all of the potential limitations and controls that may exist or the varying difficulty of any effort to exploit potential vulnerabilities. This summary also does not seek to address the patient and physician usability needs that may be impacted by and must be balanced against potential security issues. Note that for analysis, the primary areas of focus for subsystem interaction are: (i) the home monitoring system and the implantable cardiac device; (ii) the home monitoring system and the patient support network; and (iii) the physician programmer and the implantable cardiac device.

OBTAINABILITY OF VENDOR SUBSYSTEMS FROM PUBLIC SOURCES

WhiteScope was able to obtain subsystems for the four major vendors through public auction sites. As a reference point, the table below shows a snapshot of available home monitoring devices and physician programmers for purchase on eBay. Although not a specific vulnerability, the ease of obtaining implantable cardiac subsystems may aid an attacker in ecosystem exploitation. For example, if a vendor utilizes common hard-coded credentials, an attacker has the potential to glean the credentials from a subsystem purchased through a public auction site and subsequently leverage the credentials as an attack surface for multiple subsystems.

	Vendor One	Vendor Two	Vendor Three	Vendor Four
Home Monitoring Device	7	52	18	5
Physician Programmer	2	1	1	3

Number of Available Devices on eBay. Retrieved November 23, 2016.

COMMERCIAL-OFF-THE-SHELF MICROPROCESSORS

Although not intrinsically considered a vulnerability, the use of commercial-off-the-shelf microprocessors with readily available data sheets can aid an attacker in the reverse engineering process. Part numbers on ICs make it possible to find known control signals or codes in the data sheets. The codes provide searchable terms when disassembling the firmware and make it possible to find the routines that interface with specific hardware components such as flash memory. As a result, an attacker has the potential to identify critical command functions associated with subsystem operations. Additionally, the data sheets reveal the specific chip architecture. The data sheets for commercial-off-the-shelf microprocessors associated with home monitoring devices are available openly on the Internet. As a result, an attacker has the potential to identify the system architecture in order to facilitate reverse engineering. Note that it is likely that an attacker can identify functionality even without identifiable commercial-off-the-shelf microprocessors; however, their use may assist an attacker in identifying functionality more quickly and enable the use of automated hacking tools.



Home Monitoring Device Vendor One. Processor Data Sheet:
http://cache.freescale.com/files/32bit/doc/data_sheet/MC9328MX21.pdf



Home Monitoring Device Vendor Two. Processor Data Sheet:
<http://www.datasheet.hk/search.php?part=320vc5502pgf&stype=part>



Home Monitoring Device Vendor Three. Processor Data Sheet:
<http://www.alldatasheet.com/datasheet-pdf/pdf/396495/FREESCALE/MCIMX251AJM4A.html>

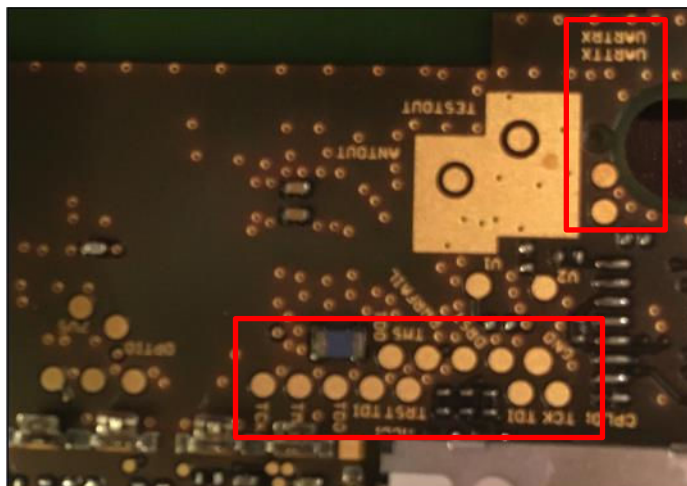


Home Monitoring Device Vendor Four. Processor Data Sheet:
<http://www.atmel.com/images/1768s.pdf>

EMBEDDED DEVICE DEBUGGING INTERFACES

Embedded devices commonly incorporate interfaces to provide in-circuit debugging functionality. Intended for functional testing, the JTAG interface can be used to acquire firmware, trace instructions, read sections of memory, capture and restore memory segments, and alter register values. JTAG interfaces were identified that permitted device interaction on home monitoring devices and physician programmers. As a result, an attacker has the potential to acquire subsystem firmware as well as pause and redirect instruction flow.

Another common debugging interface on embedded devices is the UART. The UART is typically used in debugging to allow a serial interface connection between the embedded device and PC serial port or USB. By configuring the interface parameters



Home Monitoring Device Vendor Four.

PACKED, OBFUSCATED OR ENCRYPTED FIRMWARE

Using techniques such as firmware packing, obfuscation and encryption make it much more difficult to reverse engineer firmware. Analysis of home monitoring devices for the four vendors revealed that no firmware packing, obfuscation or encryption techniques were employed. As a result, once an attacker acquires subsystem firmware, the potential exists to reverse engineer the firmware without having to decipher obfuscation or encryption.

USE OF ASCII TEXT FUNCTION NAMES AND SOFTWARE DEBUGGING

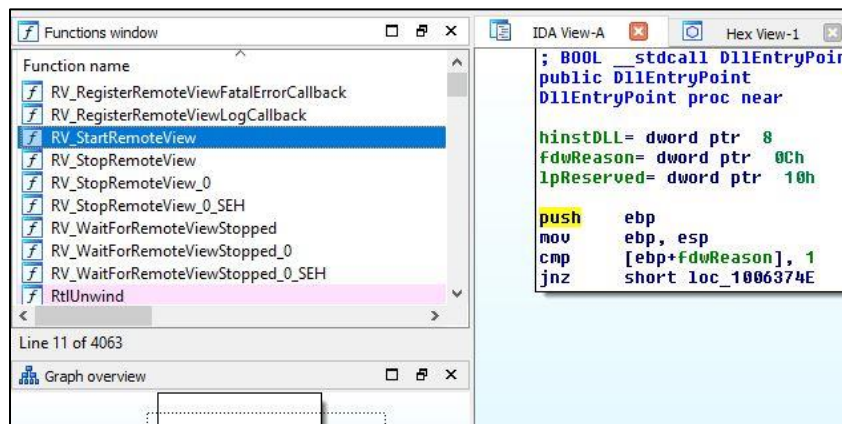
Use of ASCII text for function names provides critical clues about specific function calls that can aid reverse engineering. Additionally, software debugging symbols and source code comments can reveal functionality and critical areas of code. Analysis of the four vendor devices identified use of ASCII text for function names as well as release versions that contained software-debugging attributes. As a result, an attacker has the potential to identify critical coding sections associated with subsystem operations. Note that it is likely that an attacker can identify functionality even without ASCII text for function names and software-debugging attributes; however, minimizing their use may add a degree of difficulty for an attacker.

```

public int calculateCRC_CCITT(byte data[], int seed, int offset)
{
    int dataLen = data.length;
    byte dataWithZero[] = new byte[dataLen + 2];
    System.arraycopy(data, 0, dataWithZero, 0, dataLen);
    dataWithZero[dataLen] = 0;
    dataWithZero[dataLen + 1] = 0;
    int crcreg = seed;
    char c = '\u1021';
    for(int i = offset; i < dataWithZero.length; i++)
    {
        for(int msk = 128; msk != 0; msk >>= 1)
        {

```

Vendor One Function Example.



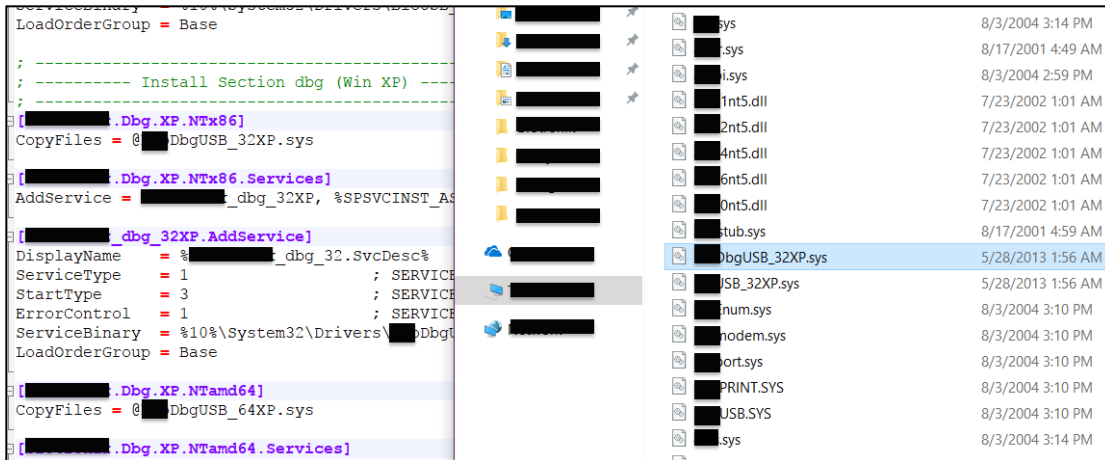
Vendor Two Function Example.

```

private boolean decryptFile(String filename)
{
    Encryption encryp = new Encryption("", new EncryptionLogger());
    try
    {
        encryp.decrypt(password, userDirectory + "patientData\\" + filename, tempFolderName);
        File dataFile = new File(tempFolderName + "data.zip");
        if(dataFile.exists())
        {
            encryp.decrypt(password, tempFolderName + "data.zip", tempFolderName);
            dataFile.delete();
        }
        copyDirectory(new File(tempFolderName), new File(desDir));
    }
    catch(IOException e1)
    {
        setErrorMessage(e1.getMessage());
        logError(e1.getMessage(), filename);
        return false;
    }
    return true;
}

```

Vendor Three Function Example.



Vendor Four Function Example.

USE OF THIRD-PARTY LIBRARIES

Software developers leverage third-party components (e.g., libraries) to help accelerate the development process. The inclusion of third-party components, however, can introduce potential vulnerabilities that often go unpatched. Analysis of the physician programmers showed the inclusion of third-party components. Additionally, multiple instances incorporated outdated and vulnerable third-party components. As a result, the potential may exist for an attacker to leverage publicly known exploits to compromise the subsystem. The numbers relating to identified third-party components usage and associated number of known vulnerabilities for physician programmers are listed below.

	Vendor One	Vendor Two	Vendor Three	Vendor Four
Number of identified third-party components	201	47	77	21
Number of vulnerable third-party components	74	39	51	10
Identified number of known vulnerabilities in third-party components	2,354	3,715	1,954	642

MAPPING FIRMWARE IMAGE INTO PROTECTED MEMORY

Mapping the firmware image into protected memory prevents the ability to overwrite or alter critical subsystem functionality while the subsystem is operating. If the firmware image is mapped into protected memory, the attacker must use a different area of memory to load malicious code. Home monitoring devices lacked an

implementation scheme that incorporates the mapping of firmware images into protected memory. As a result, an attacker has the potential to write arbitrary commands to memory and alter core system functionality.

EXTERNAL USB CONNECTIONS

Home monitoring devices included external USB connections that allowed system-level communications. By leveraging the USB connections, an attacker has the potential to traverse the file system or introduce malicious software to the home monitoring devices. Although required for some system functionality, USB connections should be locked in such a manner as to permit only authorized devices.

HARDCODED CREDENTIALS AND INFRASTRUCTURE DATA

Embedded devices often use device-to-device authentication schemes. As a consequence, credentials for authentication must be stored in some manner on the authenticating system. Analysis revealed use of hardcoded credentials on home monitoring devices for authenticating to patient support networks. In three vendors, clear text values were obtained. As a result, an attacker has the potential to use the credentials to authenticate to the patient support network. Similar to hardcoded credentials, hardcoded infrastructure data are often used in device-to-device communications. Hardcoded infrastructure data were implemented in home monitoring devices to facilitate communication with patient support networks. Infrastructure data included phone numbers and IP addresses that correspond to authentication servers for the patient support network. As a result, an attacker has the potential to identify the authentication servers for the patient support network. Due to the sensitive nature of hardcoded credentials and infrastructure data, these artifacts have been redacted from this report.

```
function download_package()
{
    # Download the devel package and the md5sum.txt file
    echo -e "\n==> Downloading $1 package using wget.\n"

    wget ftp://[REDACTED]27/$2/$1

    if [ "$?" != 0 ]; then
        echo "scp failed..."
        exit 1
    fi
}

/root/setdev.sh 1
/etc/init.d/[REDACTED]
```

Vendor One Information.

REMOTE FIRMWARE UPDATE

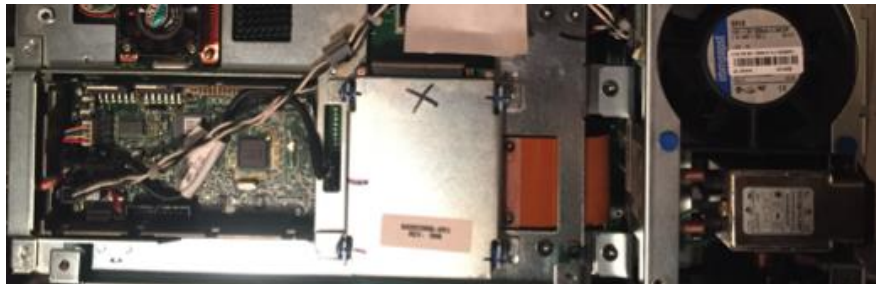
Home monitoring devices receive firmware updates via the patient support network. Home monitoring devices, however, do not necessarily validate the source of the system distributing the firmware. As a result, the potential exists to perform a man-in-the-middle attack and issue counterfeit firmware to a home monitoring device.

DIGITALLY SIGNED FIRMWARE

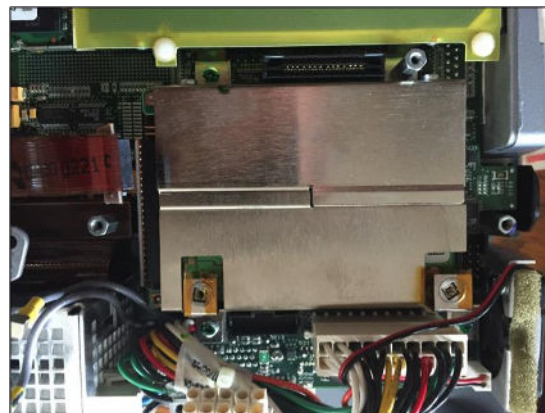
Digitally signed firmware ensures that a device will only execute authorized firmware, even if received from a non-authorized entity. Digitally signed firmware, however, was not implemented for subsystems within the implantable cardiac device ecosystem. As a result, the potential exists to load and execute counterfeit firmware on a home monitoring device.

REMOVABLE MEDIA/HARD-DRIVES

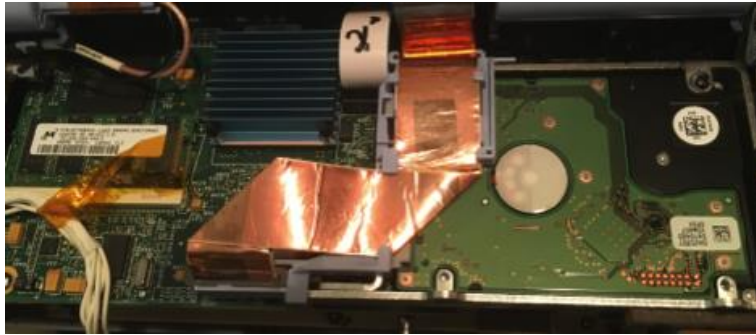
Physician programmers utilize removable media/hard-drives. As a result, an attacker has the potential to mount the removable media and extract the entire file system for the physician programmers.



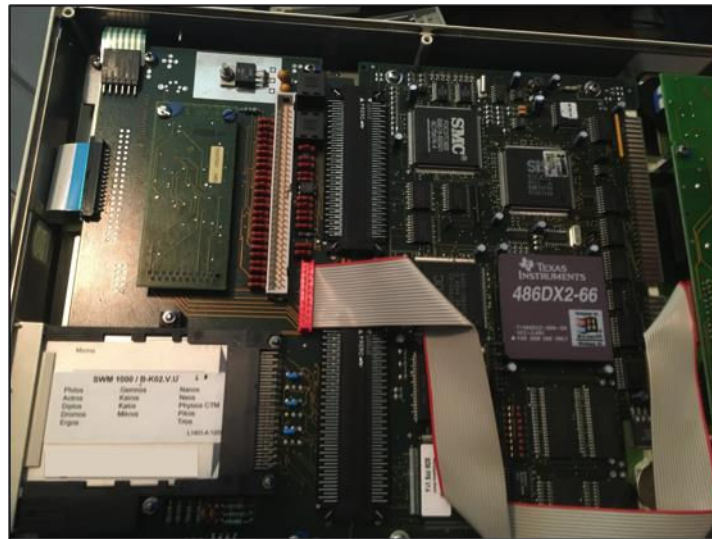
Vendor One Removable Media.



Vendor Two Removable Media.



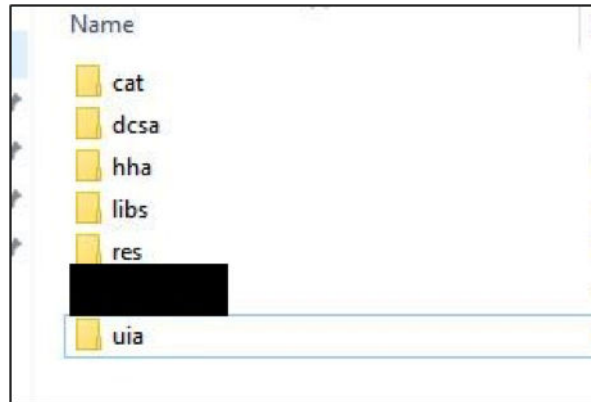
Vendor Three Removable Media.



Vendor Four Removable Media.

ENCRYPTION

File system encryption prevents unauthorized reading of subsystem data. Implementations, however, lack file system encryption for physician programmers. As a result, after the file system is extracted an attacker has the potential to read the file system.



Vendor One Unencrypted File System.



Vendor Two Unencrypted File System.

```

total 1232
drwxr-xr-x 12 root root 16384 Dec 6 2016 .
drwxr-xr-x 9 root root 16384 Dec 31 1969 ..
-rw-r--r-- 1 root root 0 Jun 11 2000 .altboot
drwxr-xr-x 18 root root 16384 Dec 6 2016 app
drwxr-xr-x 2 root root 16384 Dec 6 2016 bin
-rw-r--r-- 1 root root 516096 Jun 11 2000 .bitmap
-rw-r--r-- 1 root root 487424 Oct 12 1999 .boot
drwxr-xr-x 4 root root 16384 Dec 6 2016 etc
-rw-r--r-- 1 root root 8192 Jun 11 2000 .inodes
drwxr-xr-x 2 root root 16384 Dec 6 2016 log
drwxr-xr-x 5 root root 16384 Dec 6 2016 mau
drwxr-xr-x 2 root root 16384 Dec 6 2016 patient
-rw-r--r-- 1 root root 80 Oct 27 05:09 pgmr.cfg
-rw-r--r-- 1 root root 1 May 15 2000 sbcnt
drwxr-xr-x 2 root root 16384 Dec 6 2016 spool
drwxr-xr-x 2 root root 16384 Dec 6 2016 temp
drwxr-xr-x 2 root root 16384 Dec 6 2016 tmp
drwxr-xr-x 4 root root 16384 Dec 6 2016 usr

```

Vendor Three Unencrypted File System.

Name	Date modified	Type	Size
ADPFiles	9/5/2013 7:36 AM	File folder	
Help	9/5/2013 7:36 AM	File folder	
kernels	9/5/2013 7:36 AM	File folder	
System	9/5/2013 7:36 AM	File folder	
AppExtRes.dll	12/18/2008 9:31 A...	Application extens...	920 KB
AUM.dat	12/7/2009 8:26 PM	DAT File	1 KB
BugTracker.dll	12/18/2008 9:31 A...	Application extens...	61 KB
BugTrackerExtRes.dll	12/18/2008 9:31 A...	Application extens...	112 KB
CONSPAWN.EXE	4/3/2007 2:38 AM	Application	57 KB
Context.hlp	1/6/2009 2:29 AM	Help file	203 KB
Csh.dll	4/3/2007 2:38 AM	Application extens...	49 KB
E1usbdrv.dll	5/31/2007 5:04 PM	Application extens...	116 KB
emlname.ini	6/1/2007 11:24 AM	Configuration setti...	1 KB
.exe	12/18/2008 9:31 A...	Application	6,197 KB
Admin.exe	11/27/2008 2:30 PM	Application	57 KB
AdvancedSetup.exe	12/22/2008 11:50 ...	Application	41 KB

Vendor Four Unencrypted File System.

UNENCRYPTED PATIENT DATA

In addition to an unencrypted file system, analysis revealed two vendors do not encrypt patient data stored on the programmer hard drives. For one vendor, actual patient data was identified on the programmer obtained through the public auction site. Patient data included patient names, physicians, phone numbers, social security numbers and treatment data. This information was reported in a separate report to appropriate government agencies.

AUTHENTICATION TO CONDUCT PROGRAMMING

Physician programmers require no authentication (e.g., username/password) for programming implantable cardiac devices. As a result, access to a physician programmer provides the potential to program any supported implantable cardiac device. This finding is readily verified by powering on any of the four vendor physician programmers. Once powered on, the physician programmer operating system is loaded, and the end user can readily perform physician programmer functions.

PHYSICIAN PROGRAMMING APPLICATIONS

Physician programmers contain a separate programming application for each specific implantable cardiac device. As a result, if a security update/control is implemented for one specific application, it should also be verified and updated for all other applications on the physician programmers. If not applied to every implantable cardiac device application on the physician programmer, then the security update/control is only effective for the implantable cardiac device associated with the application that the security update/control is applied to.

DUAL USE OF RADIO HARDWARE FOR HOME MONITORING DEVICE AND PHYSICIAN PROGRAMMER

The physician programmers utilize embedded radio circuitry to transmit signals to program the implantable cardiac device. Analysis revealed that the same hardware circuitry utilized in physician programmers was used in respective home monitoring devices. As a result, the potential may exist to leverage the home monitoring device circuitry to perform the same programming functions as the physician programmer.

COMMAND WHITELISTING

Command whitelisting ensures that an implantable cardiac device only processes authorized programming functions. Configuring the implantable cardiac device to only accept authorized programming functions via an established telemetry session with a physician programmer minimizes the risk of an attacker using custom hardware or an exploited home monitoring device to maliciously program the implantable cardiac device. Analysis revealed that implantable cardiac devices lack the implementation of command whitelisting. As a result, an attacker may have the potential to spoof programming commands to the implantable cardiac device using custom hardware.

UNIVERSAL AUTHENTICATION TOKEN

Permanent authentication tokens were identified that enabled pairing of any supported home monitoring device with an implantable cardiac device. As a result, if

other security controls are not implemented an attacker may have the potential to use the universal authentication token to spoof a session with an implantable cardiac device.

SUMMARY OF FINDINGS

The table below highlights identified potential security issues associated with the implantable cardiac device ecosystem architecture. The table shows the identified security concerns mapped to evaluated vendors. Note that the identified areas highlight fundamental system architecture attributes that may give rise to potential security concerns across vendors.

	Vendor One	Vendor Two	Vendor Three	Vendor Four
Obtainability of vendor subsystems	Verified	Verified	Verified	Verified
Commercial-off-the shelf microprocessors	Verified	Verified	Verified	Verified
Debugging interfaces (JTAG/UART)	Verified	Verified	Verified	Verified
Lack of packed, obfuscated or encrypted firmware	Verified	Verified	Verified	Verified
Use of ASCII text function names	Verified	Verified	Verified	Verified
Presence of software debugging attributes	Verified	Verified	Verified	Verified
Use of third-party libraries	Verified	Verified	Verified	Verified
Lack of protected memory mapping	Verified	Verified	Verified	Verified
External USB connections that allow system-level communications	Verified	Verified	Verified	Not Identified

Hardcoded credentials	Verified	Verified	Verified	Not Identified
Hardcoded infrastructure data (e.g., dial-in phone numbers, IP addresses, server names)	Verified	Verified	Verified	Verified
RF Activation	Verified	Verified	Verified	Verified
Remote firmware update capability	Verified	Verified	Verified	Not Identified
Lack of digitally signed firmware	Verified	Verified	Verified	Verified
Removable media/hard-drive	Verified	Verified	Verified	Verified
Lack of file system encryption	Verified	Verified	Verified	Verified
Storage of unencrypted patient data	Not Identified	Verified	Verified	Not Identified
Lack of authentication to physician programmer prior to conducting implantable cardiac device programming	Verified	Verified	Verified	Verified
Use of individual applications on the physician programmer for each implantable cardiac device	Verified	Verified	Verified	Verified

The findings reveal consistency across all vendors, highlighting the inherent weaknesses in the ecosystem architecture. It is important to note that attacks that have the potential to impact patient care often require a chain of actions that bypass multiple security controls/weaknesses. Indeed, it is not common that one security

weakness alone has the potential to impact patient care. As an example, introduction of counterfeit firmware for a home monitoring device would require an attacker to obtain the firmware, reverse engineer the firmware, identify functionality within the code to modify, modify the code in a manner that creates the desired effect without breaking other subsystem functionality, repackage the firmware and distribute the firmware to home monitoring devices. As such, vendor evaluation of security controls should be in the context of patient care benefits and risk analysis while examining from a holistic perspective.

EVALUATION OF SECURITY CONTROLS

In this context, security controls relate to vendor-specific safeguards that mitigate risks associated with inherent weaknesses in the ecosystem architecture and implementation interdependencies. WhiteScope recommends that vendors perform an evaluation of their respective security controls to ensure their implementation adequately controls any inherent security risks introduced by the underlying architecture. The following questions are provided to aid vendors in evaluating their security controls against the identified architecture and implementation interdependency risks.

Are debugging interfaces (e.g., JTAG and UART) present on home monitoring devices or physician programmers? Are the interfaces or functionality disabled prior to distribution?

Is firmware on the home monitoring device packed, obfuscated and/or encrypted?

Is ASCII text used for function naming schemes in firmware code that correlates to the specific use of the system function?

Are software debugging attributes, to include debug symbols and comments, disabled in public releases of firmware/software for home monitoring devices and physician programmers?

Are third-party libraries used in software development? Are the libraries evaluated to ensure use of up-to-date releases? What processes are used to identify and update third-party libraries once the home monitoring device or physician programmer has been deployed?

Is the firmware image for the home monitoring device mapped into protected memory to prevent arbitrary writing to memory addresses?

Is the firmware image for the implantable cardiac device mapped into protected memory to prevent arbitrary writing to memory addresses?

Are external USB connections on the home monitoring device restricted to only allow communication between authorized devices? How is this security control implemented?

Are hardcoded credentials present on the home monitoring device or physician programmer? How are credentials stored? Are credentials universal in all devices?

Are hardcoded infrastructure data present on the home monitoring device or physician programmer? How are the data stored?

Does the home monitoring device implement an RF activation to initiate a session with the implantable cardiac device? Is an RF lockout procedure implemented on the implantable cardiac device to minimize the risk of continual RF activation requests that have the potential to drain the battery at a faster rate?

Do the home monitoring devices implement a remote firmware update process? What security controls are used to authenticate the source of the firmware update to the home monitoring device?

Is home monitoring device firmware digitally signed? If digital signatures are a security control that has been added, what techniques are in place to prevent loading (i.e., rollback) of a previously unsigned firmware version?

Are removable media used in the physician programmer? Is the file system on the removable media encrypted? What techniques are used to encrypt/decrypt the file system?

Are patient data stored unencrypted on the physician programmer?

Is authentication to the physician programmer required in order to program an implantable cardiac device? How is the authentication scheme implemented?

Is the same hardware used for RF communication in the home monitoring device and the physician programmer? What security controls are implemented to prevent the home monitoring device hardware from transmitting correctly formed and formatted implantable cardiac device program commands?

Are programming commands whitelisted in the implantable cardiac device to only allow device programming during an inductive telemetry initiated session?

Can the implantable cardiac device differentiate between a session established for interrogation and a session established for programming?

Is there a universal token that can be used to pair any home monitoring device/implantable cardiac device? If deemed necessary to support patient care, what other security controls protect against an attacker potentially initiating a spoofed session using the universal token?

What process is used to ensure that a security update/control applied to a physician programmer for an implantable cardiac device application is verified and applied to all other implantable cardiac device applications on the physician programmer?

Is a policy implemented that identifies customer procedures for decommissioning physician programmers? Do the procedures include methods to ensure patient data are sufficiently erased?

What processes are used to evaluate the implementation of security controls?

CONCLUSIONS

This paper provided findings from analysis performed on the implantable cardiac device ecosystem architecture and implementation interdependencies. WhiteScope identified potential areas of concern with the underlying architecture and obtained vendor devices to evaluate system implementations. The findings reveal that the inherent architecture and implementation interdependencies are susceptible to security risks that have the potential to impact the overall confidentiality, integrity and availability of the ecosystem. The findings are relatively consistent across the different vendors, highlighting the need for all vendors to perform an in-depth and holistic evaluation of implemented security controls. Given the commonality of the findings across different vendors, identification of implementation vulnerabilities as to any one vendor may expose those same vulnerabilities in other vendors and should be considered carefully before public disclosure.

By ensuring appropriate security controls are implemented, vendors can help protect against potential system compromises that may have implications to patient care.

REFERENCES

- [1] W. Burluson and K. Fu, Design Challenges for Secure Implantable Medical Devices, *Proceedings of the 49th Annual Design Automation Conference*, 2012.
- [2] D. Halperin, T. Heydt-Benjamin, B. Ramsford, 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, *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, 2008.
- [3] X. Hei, X. Du, J. Wu and F. Hu, Defending Resource Depletion Attacks on Implantable Medical Devices, *Proceedings of the 2010 IEEE Global Telecommunications Conference*, 2010.
- [4] W. Maisel and T. Kohno, Improving the Security and Privacy of Implantable Medical Devices, *The New England Journal of Medicine*, vol. 362(13), pp. 1164-1166, 2010.

[5] E. Marin, D. Singelee, F. Garcia, T. Chothia, R. Willems, B. Preneel, On the (in)Security of the Latest Generation Implantable Cardiac Defibrillators and How to Secure Them, *Proceeding of the Annual Computer Security Applications Conference*, 2016.

[6] D. Slotwiner, N. Varma, J. Akar, G. Annas, M. Beardsall, R. Fogel, N. Galizio, T. Glotzer, R. Leahy, C. Love, R. McLean, S. Mittal, L. Moricheli, K. Patton, M. Raitt, R. Ricci, J. Rickard, M. Schoenfeld, G. Serwer, J. Shea, P. Varosy, A. Verma and C. Yu, HRS Expert Consensus Statement on Remote Interrogation and Monitoring for Cardiovascular Implantable Electronic Devices, Heart Rhythm Society, Washington, DC, May 23, 2015.