Chapter 19

Attacking and Defending Healthcare Networks

By Stanislav Dashevskyi, Daniel Ricardo dos Santos and Elisa Costante

DOI: 10.1561/9781680838237.ch19

Published in *Cyber-Physical Threat Intelligence for Critical Infrastructures Security* by John Soldatos, Isabel Praça and Aleksandar Jovanović (eds.). 2021. ISBN 978-1-68083-822-0. E-ISBN 978-1-68083-823-7.

Suggested citation: Stanislav Dashevskyi, Daniel Ricardo dos Santos and Elisa Costante. 2021. "Attacking and Defending Healthcare Networks" in *Cyber-Physical Threat Intelligence for Critical Infrastructures Security*. Edited by John Soldatos, Isabel Praça and Aleksandar Jovanović. pp. 415–432. Now Publishers. DOI: 10.1561/9781680838237.ch19.

now
the essence of knowledge

The networks of Healthcare Delivery Organizations (HDOs), such as hospitals and clinics, host a multitude of special-purpose devices and protocols [1]. These include Building Automation Systems (BAS) [2, 3], which integrate physical and digital infrastructures in healthcare facilities – such as lighting, video surveillance, power supply, fire detection, and physical access control – as well as connected medical devices [4], which can communicate with enterprise systems to optimize patient care.

These devices increasingly rely on IP-based networks and often share the network with general-purpose IT equipment [5]. Hence, malicious actors can exploit vulnerabilities on protocols and devices to launch attacks on healthcare facilities [6], which can lead to financial losses or even harm patients, staff, and other building occupants. Attacks on BAS can, e.g., cause blackouts by damaging power systems or grant access to restricted areas by tampering with physical access control [7]. On the other hand, direct attacks against medical devices can affect the health or

the quality of care provided to patients by, e.g., tampering with diagnostics and vital readings [8].

Even though their communications are IP-based, devices in HDOs use domain-specific (and often proprietary) protocols [9], which are mostly ignored by traditional Intrusion Detection Systems (IDS). Thus, the detection of complex cyberattacks on these systems requires dedicated tools to parse and analyze their network traffic.

This chapter reviews some cybersecurity challenges observed in healthcare networks (Section 19.1), discusses a set of attacks targeting medical devices that leverage insecure protocols (Section 19.2), and describes an innovative network-based IDS that relies on in-depth protocol parsing and is specifically designed to protect healthcare networks (Section 19.3). This IDS recognizes the different types of traffic on the network (e.g., BAS and medical), parses the contents of messages, and combines both signature- and anomaly-based detection to identify a wide range of attacks. Section 19.4 concludes this chapter.

19.1 Background: Security Challenges in Healthcare Networks

As discussed in [5], healthcare networks differ from typical enterprise IT networks based on the types of devices deployed and the protocols they use. These networks host not only very sensitive connected medical devices that may be attached directly to patients, but also IT equipment used to store and process patient health and financial information, as well as more diverse Internet of Things (IoT) and Operational Technology (OT) devices that have distinct purposes, such as building automation and control.

The security challenges of real-world healthcare networks were analyzed in detail in [5, 9, 10], with examples of potential threats including financially motivated cyberattacks and cyberattacks targeting patient safety. Among the main findings of those studies are the reliance of healthcare networks on insecure protocols and their lack of proper network segmentation. Below, we review these challenges.

19.1.1 Insecure Protocols

Building automation and medical devices in HDOs transmit data on the network using either standard protocols or proprietary ones, which are developed by vendors for use within their device ecosystems.

The most popular BAS protocols include BACnet [11] and RTP/RTSP [7]. BACnet is a general purpose, multi-stack network protocol specifically devised to control several building automation systems such as HVAC, lighting, and access control. BACnet is by far the most widely used network protocol in building automation systems. RTP is used for real-time transfer of streaming data, such as audio or video. RTSP is a text-based protocol, with a syntax that resembles HTTP, supporting commands such as PLAY, PAUSE, and TEARDOWN to establish and control media sessions between client and server endpoints, such as for instance IP cameras and NVRs.

The most popular healthcare protocols include standards such as HL7, DICOM, POCT01, LIS02, as well as proprietary ones, such as GE RWHAT and Philips Data Export [5]. HL7 is the most widely used interoperability and data exchange protocol in medical networks, which allows for the exchange of patient, clinical, and administrative information. DICOM defines both the format for storing medical images and the communication protocol used to exchange them. DICOM is implemented by all major vendors of devices involved in medical imaging processes, such as diagnostic workstations, storage servers, and medical printers. POCT01 and LIS02 are used for point-of-care testing and laboratory testing devices, respectively. These protocols can issue test orders to devices and are used by the devices to communicate the results of tests back to a data management system. The proprietary protocols Philips Data Export and GE RWHAT are used to control patient monitors of their respective vendors and to communicate the vital readings of patients to a central monitoring system.

These protocols, as well as others used by building automation and medical devices, often lack support for encryption and authentication or do not enforce their usage [5, 7, 12]. This is, among other reasons, because they are designed to accommodate resource constraints of embedded devices and assume that communication happens in internal "closed" networks that are not accessible to attackers. Some standards (such as HL7, DICOM and POCT01) cite the possibility of encrypting the transmitted data but leave the choice of implementation to individual deployments (sometimes, assuming, that encryption happens at a lower layer, e.g., by using TLS). As discussed in [5] this means that this communication is often done in cleartext.

The consequence of relying on insecure protocols is that a well-positioned attacker can sniff sensitive data, tamper with the communication of devices or inject malicious data, thus allowing for physical intrusions and harm to healthcare facility occupants, such as patients, guests, and staff. The consequences of insecure medical protocols are even more critical, since the data being transmitted is often sensitive and the effects of tampering with commands issued by medical devices can be dire.

19.1.2 Improper Network Segmentation

Network segmentation is a fundamental measure to limit the attack surface in networks by isolating or limiting access to critical devices and grouping those devices by network function. Segmentation is often achieved by a combination of techniques at network layers 2 and 3, including Virtual Local Area Networks (VLANs).

Improper network segmentation, with segments that mix sensitive and vulnerable devices, allows for attackers to move laterally in a network, thus increasing the potential impact of an attack. The study in [5] shows that less than 20% of medical devices are deployed in a VLAN and that 86.5% of HDOs have medical devices outside of VLANs. In addition, the study [5] identified several VLANs at real hospitals hosting a combination of medical devices and other types of devices, such as medical imaging modalities or blood monitors and IP cameras, thus undermining the segmentation benefits that a VLAN may provide.

The consequence of improper network segmentation is that an attacker with access to less privileged or sensitive assets may be able to attack or move laterally to a more privileged or sensitive asset.

19.2 Attacking Healthcare Networks: Exploiting Insecure Protocols

To demonstrate how the challenges described in Section 19.1 can lead to the exploitation of a healthcare network, we set up a small lab where we could reproduce the goals of a malicious actor.

The lab is depicted in Figure 19.1. On the clinical side, there are two medical devices: a patient monitor and a blood analyzer. On the BAS side, there is an IP camera. On the IT side, the lab contains a Network Video Recorder (NVR) that displays the video footage from the IP camera using the iSpy[1] software, a Central Monitoring Station (CMS) that shows the real-time readings of the patient monitor using the ixTrend Express software,[2] and a Laboratory Information System (LIS) that stores test results from the blood analyzer. Since we did not find a suitable solution for LIS, we implemented a simple LIS02 server using Python ASTM,[3] and a POCT01 server according to the communication specifications of the blood analyzer [13].

1. https://www.ispyconnect.com/
2. https://www.ixellence.com/index.php/en/home/17-default-en/products
3. https://pypi.org/project/astm/

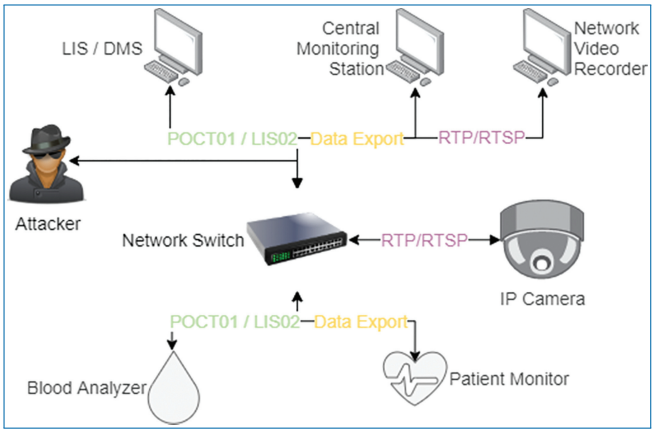

Figure 19.1. Lab setting used to demonstrate attacks.

All devices on this lab are connected to the same network switch in the center, without network segmentation, and their communications are in clear-text – representing the challenges we discussed in Section 19.1.

The Figure also shows an attacker that has local access to the network. In a real setting, this access could be obtained via, e.g., network sockets in patient rooms [8] or by exploiting Internet-connected devices [7]. We assume that the attacker can sniff and, when necessary, modify packets in the network, essentially acting as a man-in-the-middle (MITM), which can be achieved via, e.g., ARP poisoning using ettercap.[4]

We demonstrated in [7] how an attacker can exploit insecure video streaming protocols (such as RTP and RTSP) to prevent the NVR from displaying the correct footage to an operator, and also how an attacker can exploit an Internet-connected IP camera to gain external access to a building automation network. These attack types are critical, especially for healthcare facilities, where a compromise of the video surveillance system could be only the first step of a physical intrusion.

Below, we describe attacks that demonstrate how a malicious actor can compromise the safety of patients in an HDO by leveraging the insecure communications of medical devices.

19.2.1 Dumping Test Results

The goal of this attack is to intercept test results sent from the blood analyzer to the LIS, although this attack (at least the passive variant) could be reproduced with any two devices communicating over unencrypted and unauthenticated POCT01.

4. https://www.ettercap-project.org/

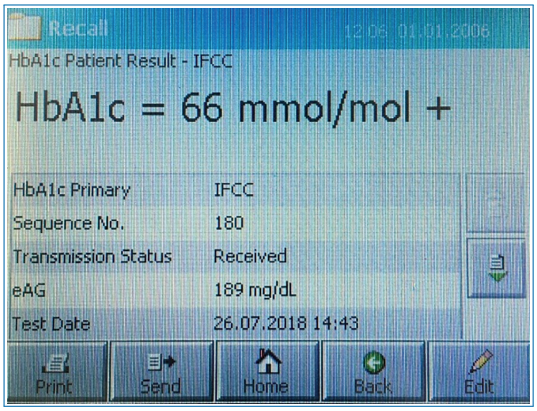

Figure 19.2. HbA1c test result shown on the screen of the blood analyzer.

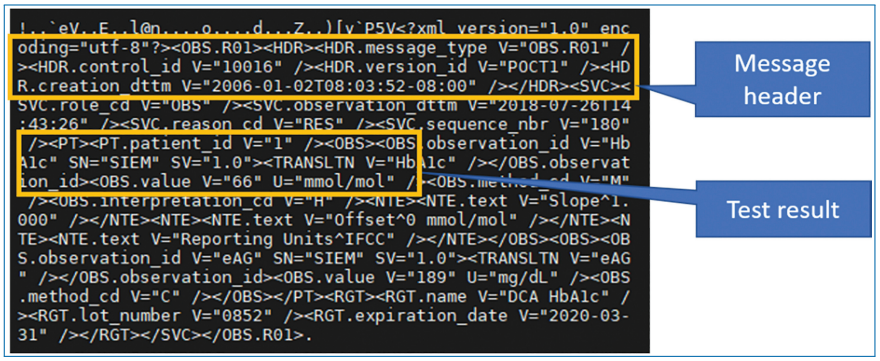

Figure 19.3. Details of the POCT01 data payload transmitting a test result to the LIS.

An example of a blood test result from the analyzer is shown in Figure 19.2. The Figure shows the result of an HbA1c test, which measures the level of blood sugar over a period of weeks and is routinely done for patients with diabetes. The result shown in Figure 19.2 is 66 mmol/mol, which is indicative of diabetes.

When the device operator chooses to send a test result to the LIS via the POCT01 protocol, and there exists an established synchronous POCT01 conversation between the blood analyzer and the LIS, a data payload such as the one shown in Figure 19.3 is generated and sent over the network. The payload contains a message header that specifies the message type ("OBS.R01" stands for a test result sent from the blood analyzer) and the test result creation timestamp. Further, the payload lists the patient id and the test result value.

Since the data is transmitted in cleartext, attackers can passively intercept test results sent over by operators by simply sniffing the network traffic and examining the POCT01 packets that contain the "OBS.R1" message type in the message header.

However, attackers can also actively intercept test results by bringing into the hospital rogue devices that can serve as fake LIS servers. Due to the lack of traffic encryption, these devices can then hijack communications between a POCT device and a legitimate LIS server.

As a proof-of-concept, we have implemented a rogue LIS server, according to the device-specific POCT01 communication protocol implemented in the blood analyzer [13]. We first perform an ARP cache poisoning attack, so that the blood analyzer is forced to communicate with the rogue server. Once the device sends the hello message ("HEL.R01"), the server responds with an ack message ("ACK.R01"), requests pending tests results ("REQ.R01"), obtains the results, and, after a short conversation sequence (detailed in [13]) establishes a continuous conversation mode with the blood analyzer. In this mode, all further test results will be sent directly to the rogue LIS server. Moreover, the blood analyzer will accept a limited set of commands from the server, such as to update the list of the device's operators ("OPL.R01").

19.2.2 Changing Test Results

The goal of this attack is to tamper with a test result sent from the blood analyzer to the LIS via the LIS02 protocol, although this attack could be reproduced with any two devices communicating over unencrypted and unauthenticated LIS02 or POCT01.

When the operator chooses to send a test result (the same one seen in Figure 19.2) to the LIS via the LIS02 protocol, a data payload such as the one shown in Figure 19.4 is generated and sent over the network.

Notice that the data payload contains a header with some information about the device issuing the result, a timestamp, detailed test results, following with a checksum. (For a complete reference on the contents of a LIS02 packet, see [14]).

When the LIS server receives the packet, it displays the results as shown in Figure 19.5 and stores it internally.

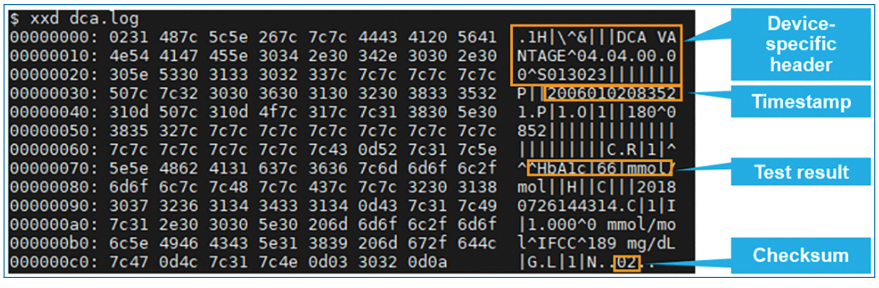

Figure 19.4. Details of the LIS02 data payload transmitting a test result to the LIS.

```
----------------------------
RECEIVED!
----------------------------
HEADER: ['H', [[None], [None, '&']], None, None, ['DCA VANTAGE', '04.04.00.00', 'S0113023'],
 None, None, None, None, None, None, 'P', None, '2006010208352']

PATIENT: ['P', '1']

ORDER: ['O', '1', None, ['180', '0852']], None, None, None, None, None, None, None, None, No
ne, None, None, None, None, None, None, None, None, None, None, None, None, 'C']

RESULT: ['R', '1', [None, None, None, 'HbA1c'], '66', 'mmol/mol', None, 'H', None, 'C', None
, None, '20180726144314']

COMMENT: ['C', '1', 'I', ['1.000', '0 mmol/mol', 'IFCC', '189 mg/dL', 'G']]

TERMINATOR: ['L', '1', 'N']
```

Figure 19.5. Test result received by the LIS.

```
----------------------------
RECEIVED!
----------------------------
HEADER: ['H', [[None], [None, '&']], None, None, ['DCA VANTAGE', '04.04.00.00', 'S0113023'],
 None, None, None, None, None, None, 'P', None, '2006010208357']

PATIENT: ['P', '1']

ORDER: ['O', '1', None, ['180', '0852']], None, None, None, None, None, None, None, None, No
ne, None, None, None, None, None, None, None, None, None, None, None, None, 'C']

RESULT: ['R', '1', [None, None, None, 'HbA1c'], '41', 'mmol/mol', None, 'H', None, 'C', None
, None, '20180726144314']

COMMENT: ['C', '1', 'I', ['1.000', '0 mmol/mol', 'IFCC', '189 mg/dL', 'G']]

TERMINATOR: ['L', '1', 'N']
```

Figure 19.6. LIS displaying changed test result due to an attack.

When an attacker wants to tamper with this flow and send incorrect results to the LIS, they must do the following: create a man-in-the-middle, drop the original data packet, create a new packet with a modified test result, compute the new checksum, insert it into the new packet and send it. This can be easily achieved with a tool such as ettercap and a custom filter that modifies test results and checksums.

The result of a modified packet received by the LIS server is shown in Figure 19.6. This would cause the LIS to store an incorrect result. In this example, the attacker has changed the results of a diabetic patient to a normal test result (41 mmol/mol). The opposite could just as easily be done.

19.2.3 Disconnecting a Patient Monitor

The goal of this attack is to close an ongoing connection between the patient monitor and the CMS, so that the medical staff remotely monitoring a patient loses the real-time information about vital readings.

This is achieved by issuing an "Association Abort" command that is available on the Philips Data Export proprietary protocol [15]. The attacker can spoof an abort message and make the CMS believe that the monitor wants to close the ongoing connection, thus causing a denial of service. To do so, the attacker simply has to craft a packet containing a payload with the abort message and send it to the UDP

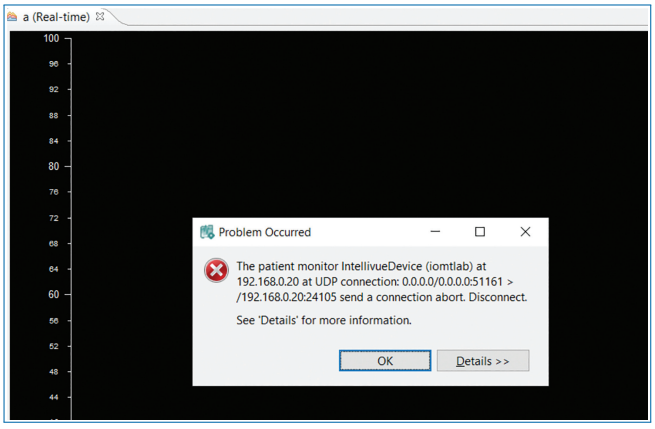

Figure 19.7. CMS displaying the result of an Association Abort message.

port 24105, which is used by default the Data Export protocol (or any other port that is used in the connection between the monitor and the CMS, which can be learned by sniffing the traffic).

The result of the attack can be seen in Figure 19.7, where the CMS is shown displaying an error message informing the user that the monitor has closed the connection and stopped sending data to the CMS.

19.2.4 Changing a Patient's Vital Readings

The goal of this attack is to tamper with the vital readings sent from the patient monitor to the CMS, so that the medical staff remotely monitoring a patient sees incorrect real-time information about vital readings.

This is achieved by modifying on-the-fly the Data Export packets sent from the monitor to the CMS. To do so, the attacker can again use ettercap and create a filter that replaces the real-time vital readings with a desired value.

The only challenge in this case is to understand at which offset in the packets the vital readings are encoded, since Data Export is a binary protocol. This information can be obtained from the Data Export manual [15] for the pulse rate (which we use in the examples below), blood pressure and oxygen saturation.

Searching for the values 0x4822 and 0x0aa0 (indicating the fields containing the pulse rate values) on captured traffic between the monitor and the CMS, we find what is shown in the UDP packet on Figure 19.8, where the bytes with values 48 22 indicate to the CMS that a pulse value is incoming and the bytes with values 0a a0 indicate the unit (beats per minute). Finally, the last two bytes encode the actual value of the pulse observed in the monitor, which in this case is 50 in hexadecimal or 80 in decimal. Therefore, we can calculate the offset of the byte we want to change (the one with value 50).

```
02e0  42 28 00 00 00 00 09 24  00 04 00 02 48 22 09 27   B(·····$ ····H"·'
02f0  00 10 00 0e 00 50 00 75  00 6c 00 73 00 65 00 20   ·····P·u ·l·s·e·
0300  00 00 09 11 00 02 00 06  09 50 00 0a 48 22 00 01   ········ ·P·H"··
0310  0a a0 00 00 00 50 83 a7  00 07 00 4e 09 21 00 02   ·····P·· ···N·!··
0320  83 a7 09 2f 00 04 00 01  00 06 09 3f 00 0c 00 00   ···/···· ···?····
0330  20 00 00 01 42 28 00 00  00 00 09 24 00 04 00 02    ···B(·· ···$····
0340  4b b0 09 27 00 10 00 0e  00 50 00 65 00 72 00 66   K··'···· ·P·e·r·f
0350  00 20 00 20 00 00 09 11  00 02 00 06 09 50 00 0a   · · ···· ·····P··
0360  4b b0 00 00 02 00 ff 00  00 17                     K······ ··
```

Figure 19.8. Packet with patient's pulse rate transmitted from the patient monitor.

Figure 19.9. Normal pulse rate reading on the patient monitor.

Once this offset is discovered, the attacker can create an ettercap filter to extract the right packet containing the patient data and modify the pulse value of the patient to their desired value (e.g., 0 to simulate a patient flatlining, or a rapid succession of high and low numbers to simulate an arrythmia condition) and forward it to the CMS to display this information.

Figure 19.9 shows the actual reading on the patient monitor, which is a normal pulse of 83. Figure 19.10 shows the result of the flatlining attack as seen by staff on the CMS. Notice that the pulse suddenly drops from a normal range between 70 and 80 to 0. Similar attacks could be implemented to change the oxygen saturation and blood pressure readings, for instance.

19.3 Defending Healthcare Networks: Intrusion Detection

As demonstrated in the previous Section, complex attacks targeting healthcare networks may leverage subtle changes in domain-specific protocols to achieve malicious goals.

This Section describes a Network Intrusion Detection System (NIDS) designed to detect attacks targeting healthcare networks, such as the ones described in Section 19.2, by parsing domain-specific network protocols and combining signature-based with anomaly-based detection. The NIDS was implemented on

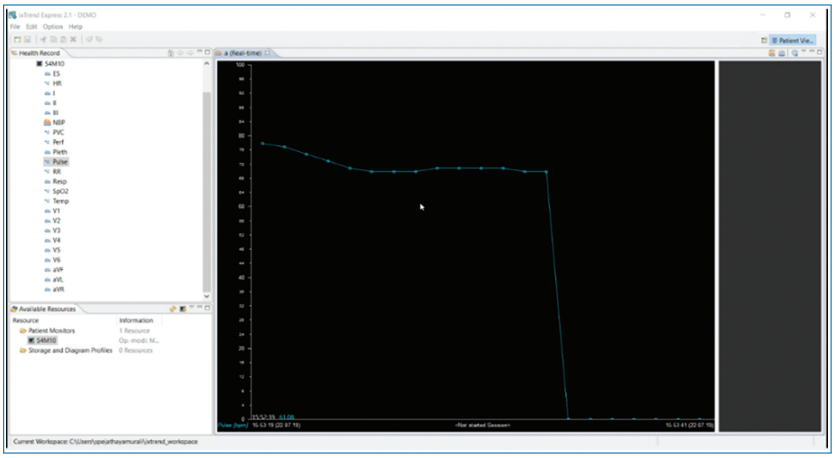

Figure 19.10. Result of a flatlining attack shown in the CMS.

top of Forescout eyeInspect,[5] a commercially distributed OT-focused IDS, as part of the SAFECARE[6] project.

To avoid disrupting the operational continuity of healthcare networks, due to their criticality for the safety of patients and building occupants, the NIDS is based on passive monitoring detection modules. This means that the NIDS does not inject any traffic into the monitored network, it only observes the traffic generated by other devices.

Figure 19.11 shows the architecture of the NIDS, which encompasses the components described in Sections 19.3.1–19.3.3.

19.3.1 NIDS Sensor

The core of the NIDS (called the sensor) is a network sniffer that intercepts and dissects the traffic passing on the wire using deep packet inspection. A sensor is connected to a pre-configured port of a switch in the healthcare network that mirrors all traffic going through a network segment.

The sensor can dissect several standard and proprietary protocols commonly used in building automation and medical systems (such as the ones mentioned in Section 19.1). This component provides the evidence extracted from raw network traffic, which is then fed into the detection modules for raising security alerts. For some protocols that support file transfers (e.g., SMB), the sensor can also dissect the files and make them available in the monitoring interface.

5. https://www.forescout.com/platform/eyeinspect/

6. https://www.safecare-project.eu/

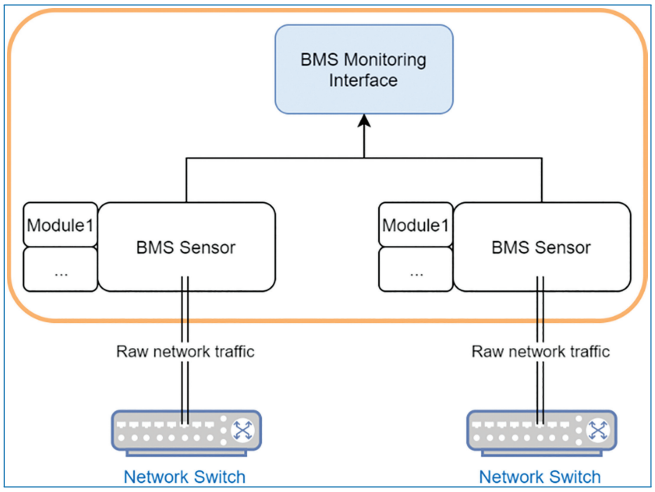

Figure 19.11. BAS threat detection system architecture.

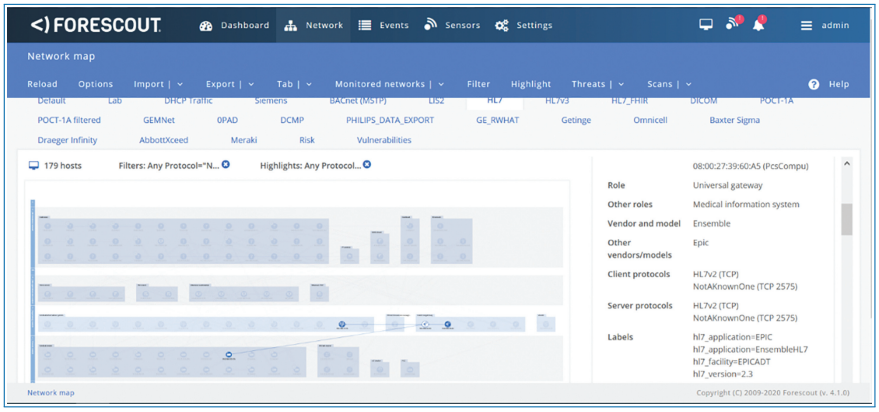

Figure 19.12. Network Map view.

19.3.2 NIDS Monitoring Interface

The monitoring interface receives, aggregates, and displays the data coming from the sensors placed in the healthcare network. This interface provides the user with actionable information about the assets present in the network and is responsible for sending the alerts raised by the detection modules to third-party systems.

Figure 19.12 shows the Network Map view, where the user can quickly identify all the assets in the network and how they communicate. The Network Map has configurable tabs where the user can specify filters to display only assets of interest, such as the protocol-based filters shown at the top (e.g., BACnet, HL7, DICOM, etc.).

In the network map, assets are grouped by their roles, which are automatically identified based on the communications and parsed properties of each asset (see [16] for a description of how this can be achieved). If the user clicks on one node of the map, the details of the corresponding asset are shown on the right (such as the Epic HL7 Gateway shown in the Figure). All the information displayed is obtained from passive monitoring of the network traffic and parsing the protocols used by the asset (in the example, mainly HL7).

For forensic analysis, the NIDS also keeps an activity log of devices, which contains not only related alerts, but also Host Change logs (such as new protocols, ports, etc.) and Network Logs (such as DNS requests). Another forensic functionality of the Monitoring Interface is to run checks for stored network logs based on new Indicators of Compromise, such as blacklisted IP addresses and file hashes, that are released frequently. That allows a user to know if the network was silently attacked in the past (e.g., for espionage or data exfiltration).

19.3.3 Detection Modules

The detection modules are the passive detection engines incorporated in the NIDS sensor. They are responsible for analyzing the parsed traffic from the sensors and detecting known attacks or anomalies that represent potentially malicious behavior. For each alert raised, a short packet capture before and after the suspicious activity can be stored by the sensor to facilitate post-incident forensic analysis. Each module is fully configurable and can be turned off if needed. Below, we describe each detection module of the NIDS.

19.3.3.1 Signature-based detection

The signature-based module provides several pre-configured checks and controls to detect weaknesses and threats at an early stage and offers intelligence about the cause and remediation of a detected problem. These signatures are flexible, and each individual check can be enabled or disabled to accommodate most of the use cases required by an HDO. The checks are divided into three categories:

- **Networking checks** detect device and network misconfigurations, such as hosts not receiving responses or connectivity issues.
- **Operations checks** detect problems and threats to the building automation operations, such as malfunctioning or misbehaving devices or the use of potentially dangerous operations (e.g., restart/reset commands).
- **Security checks** detect security threats and vulnerabilities, such as the use of insecure protocols or protocol versions (e.g., TELNET or SSHv1), exploits of known vulnerabilities, and indicators of compromise.

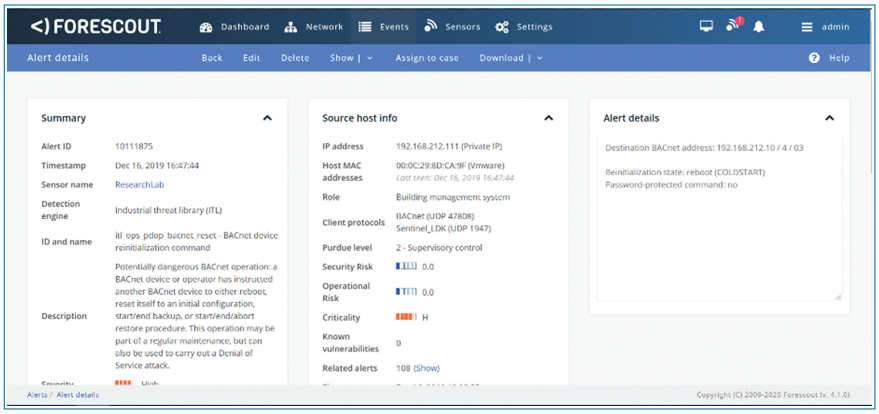

Figure 19.13. Alert raised by dangerous BACnet operation.

Figure 19.13 shows an alert raised by the "BACnet device reinitialization command" check. Notice that the alert contains details about the event (such as a timestamp, description, severity, and the assets involved) that allow the user to investigate its relevance.

19.3.3.2 Anomaly-based detection

The anomaly-based detection engine is used to model network communications within a local network environment, i.e. a network with a limited number of (known) hosts communicating with each other. The anomaly-based engine can model network communications by the following features that span across the network protocol stack: IP addresses, L4 (transport layer) protocol, L4 ports, L7 (application layer) protocol, and L7 message groups (e.g., read, write, delete).

Modeling is done by means of communication rules. A communication rule defines an action to be performed by the engine when the observed network communication matches the IP addresses, L4 protocol, L4 ports, L7 protocol and L7 message groups specified in the rule.

The most common actions which can be defined are: *allow* and *alert*. If the action is *allow*, the rule defines a whitelisted communication. If the action is *alert*, the rule defines a blacklisted communication and an alert is raised when the communication is detected. The anomaly-detection module can be set in learning mode in order to automatically detect the rules from network traffic. One rule is created for each combination of source IP address, destination IP address, L4 protocol, destination L4 port and L7 protocol. When set into detecting mode, the anomaly-based engine checks the network communications for a matching rule and reacts according to the specified action. The rules are checked at different stages of a network communication.

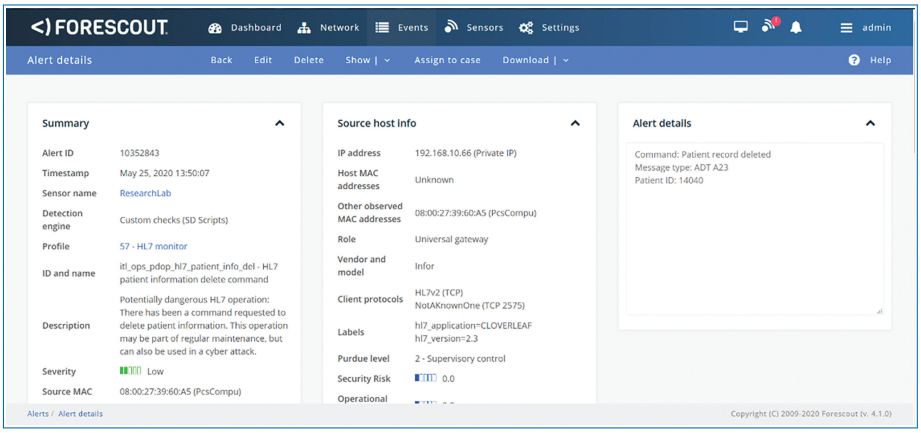

Figure 19.14. Alert raised by HL7 delete patient data command.

Anomaly-based detection can also be tailored to detect suspicious behavior using specific protocols. For instance, an alert can be raised when someone tries to change the port where an IP camera is streaming via the RTSP protocol (which was used in the footage replay attack described in [7]). Similarly, an alert can be raised when an unknown host streams to an NVR, which is another step taken in the footage replay attack. Another example is an alert raised when an HL7 "delete patient data" command is seen on the network. This is anomaly-based because the alert depends on the number of "delete" messages seen on the network, since a single message is normal, but a quick succession of such messages may indicate an attacker trying to erase data in a hospital.

19.3.3.3 Other detection modules

Other detection modules in the NIDS include specialized techniques for the detection of malformed packets (i.e. packets that do not conform to a protocol's specification and may be attempting to exploit a vulnerability), TCP port scans (which are typically an initial step of an attack) and man-in-the-middle attacks (which can be used by attackers to tamper with communications, as demonstrated in Section 19.2).

19.3.4 Interconnections

The objective of SAFECARE is to bring together advanced technologies of physical and cyber security to manage combined cyber and physical threats, their interconnections, and potential cascading effects. The project focuses on health service infrastructures and works towards the creation of a comprehensive protection system called the SAFECARE platform. Within the SAFECARE platform, the NIDS

described in this Section communicates directly with two other components: the Cyber Threat Monitoring System (CTMS) and the Advanced Malware Analyzer (AMA).

The CTMS integrates information acquired by the different detection systems composing the SAFECARE cyber-security solution into an incident which is correlated with the physical security information and stored in a central database. The alerts generated by the NIDS are sent to the CTMS using the Syslog protocol [17]. An example of a Syslog message representing an alert raised when resetting a building controller using the BACnet protocol (as shown in Figure 19.13) is as follows:

> CEF:0|SAFECARE|NIDS|BACnet Device Reinitialization Command|severity=HIGH|cat=alert alert_type=bacnet_device_reset id=1 smac=00:0a:0a:0a dmac=00:0b:0b:0b src=192.168.1.1 src_risk=HIGH dst=192.168.1.2 dst_risk=MEDIUM src_port=47809 dst_port=47810 l4proto=udp l7proto=bacnet module=SignatureModule timestamp= 2019-10-25T10:34:24.461+02:00 msg={Potentially dangerous BACnet operation: a BACnet device or operator has instructed another BACnet device to either reboot, reset itself to an initial configuration, start/end backup, or start/end/abort restore procedure. This operation may be part of a regular maintenance but can also be used to carry out a Denial of Service attack.}

The AMA is responsible for detecting malicious files in the networks monitored by SAFECARE. As discussed in Section 19.3.1, the NIDS sensor is capable of dissecting files from raw network traffic of protocols, such as SMB, and making them available to the monitoring interface for analysis against known malware hashes or malicious content, so that the transfer of malicious files within a network will raise security alerts.

An important case of this functionality is related to DICOM files, which are widely used in the healthcare domain and may embed executable files while still being valid images (a vulnerability known as PEDICOM[7]). A YARA[8] rule is used by the NIDS sensor to detect the transfer of DICOM files (searching the string "DICM" starting at byte 128) that embed potentially malicious executable information (searching for the DOS MZ executable header, represented by the hexadecimal value 5A4D), dissect the file, and forward it to a pre-configured instance of the AMA. The AMA can then examine the file and decide whether it is malicious or not.

7. https://github.com/d00rt/pedicom

8. https://github.com/VirusTotal/yara

19.4 Conclusion

This chapter reviewed two cybersecurity challenges observed in healthcare networks (the use of insecure protocols and improper network segmentation), discussed a set of four attacks targeting medical devices that leverage those challenges, and described an innovative network-based IDS that relies on in-depth protocol parsing to protect healthcare networks from such cyber-attacks.

Moreover, the chapter highlighted the fact that to properly defend healthcare networks from complex attacks, an IDS must be able to understanding domain-specific protocols used by devices such as building automation and medical equipment. Such an IDS is a clear step forward to achieve the SAFECARE objective of detecting and managing cyber-threats in healthcare.

Acknowledgements

This work has received funding from European Union's H2020 research and innovation programme under SAFECARE Project, grant agreement no. 787002.

References

[1] G. O'Brien, S. Edwards, K. Littlefield, N. McNab, S. Wang and K. Zheng, "Securing Wireless Infusion Pumps In Healthcare Delivery Organizations," NIST, 2018.

[2] P. Domingues, P. Carreira, R. Vieira and W. Kastner, "Building automation systems: Concepts and technology review," *Computer Standards & Interfaces*, vol. 45, no. 1, pp. 1–12, 2016.

[3] W. Kastner, G. Neugschwandtner, S. Soucek and H. Newman, "Communication systems for building automation and control," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1178–1203, 2005.

[4] A. Gatouillat, Y. Badr, B. Massot and E. Sejdic, "Internet of Medical Things: A Review of Recent Contributions Dealing With Cyber-Physical Systems in Medicine," *IEEE IoT Journal*, vol. 5, no. 5, pp. 3810–3822, 2018.

[5] G. Dupont, D. dos Santos, E. Costante, J. den Harotg and S. Etalle, "A Matter of Life and Death: Analyzing the Security of Healthcare Networks," in *IFIP SEC*, 2020.

[6] T. Mundt and P. Wickboldt, "Security in building automation systems – a first analysis," in *Proceedings of Cyber Security*, 2016.

[7] D. dos Santos, M. Dagrada and E. Costante, "Leveraging Operational Technology and the Internet of Things to Attack Smart Buildings," *J Comput Virol Hack Tech*, 2020.
[8] Y. Mirsky, T. Mahler, I. Shelef and Y. Elovici, "CT-GAN: Malicious Tampering of 3D Medical Imagery using Deep Learning," in *USENIX Security*, 2019.
[9] D. Foo Kune, K. Venkatasubramanian, E. Vasserman, I. Lee and Y. Kim, "Toward a Safe Integrated Clinical Environment: A Communication Security Perspective," in *MedCOMM*, 2012.
[10] ISE, "Securing Hospitals: A Research Study and Blueprint," 2016. [Online]. Available: https://www.securityevaluators.com/hospitalhack/.
[11] ASHRAE, "BACnet – A Data Communication Protocol for Building Automation and Control Networks," 2016. [Online].
[12] P. Ciholas, A. Lennie, P. Sadigova and J. Such, "The Security of Smart Buildings: a Systematic Literature Review," 2019. [Online]. Available: https://arxiv.org/abs/1901.05837.
[13] Siemens, DCA Vantage™ Analyzer Host Computer Communications Link, 2011.
[14] CLSI, "LIS02," [Online]. Available: https://clsi.org/standards/products/automation-and-informatics/documents/lis02/.
[15] Philips, "Data Export Interface Programming Guide," [Online].
[16] D. Fauri, M. Kapsalakis, D. dos Santos, E. Costante, J. den Hartog and S. Etalle, "Role Inference + Anomaly Detection = Situational Awareness in BACnet Networks," in *DIMVA*, 2019.
[17] I. Eaton, "The Ins and Outs of System Logging Using Syslog," 2003. [Online]. Available: https://www.sans.org/reading-room/whitepapers/logging/paper/1168.