Article

Protected Network Architecture for Ensuring Consistency of Medical Data through Validation of User Behavior and DICOM Archive Integrity

Shamil Magomedov * and Artem Lebedev

Department of Intelligent Information Security Systems, MIREA—Russian Technological University, 119454 Moscow, Russia; lebedev_a@mirea.ru

* Correspondence: magomedov_sh@mirea.ru

Abstract: The problem of consistency of medical data in Hospital Data Management Systems is considered in the context of correctness of medical images stored in a PACS (Picture Archiving and Communication System) and legality of actions authorized users perform when accessing MIS (Medical Information System) facilities via web interfaces. The purpose of the study is to develop a SIEM-like (Security Information and Event Management) architecture for offline analysis of DICOM (Digital Imaging and Communications in Medicine) archive integrity and users' activity. To achieve amenable accuracy when validating DICOM archive integrity, two aspects are taken into account: correctness of periodicity of the incoming data stream and correctness of the image data (time series) itself for the considered modality. Validation of users' activity assumes application of model-driven approaches using state-of-the-art machine learning methods. This paper proposes a network architecture with guard clusters to protect sensitive components like the DICOM archive and application server of the MIS. New server roles were designed to perform traffic interception, data analysis and alert management without reconfiguration of production software components. The cluster architecture allows the analysis of incoming big data streams with high availability, providing horizontal scalability and fault tolerance. To minimize possible harm from spurious DICOM files the approach should be considered as an addition to other securing techniques like watermarking, encrypting and testing data conformance with a standard.

Keywords: DICOM archive; data validation; user behavior analytics; distributed systems; network traffic interception

Citation: Magomedov, S.; Lebedev, A. Protected Network Architecture for Ensuring Consistency of Medical Data through Validation of User Behavior and DICOM Archive Integrity. *Appl. Sci.* **2021**, *11*, 2072. <https://doi.org/10.3390/app11052072>

Academic Editor: Federico Divina

Received: 23 January 2021

Accepted: 22 February 2021

Published: 26 February 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Today, healthcare facilities are vast ecosystems made up of a large number of network devices, equipment and systems that often require connection to external systems. Medical data are very sensitive to change, and poses a real threat to the health and life of patients. One does not need to have special skills to become familiar with the potential vulnerabilities that a healthcare facility may face. Therefore, the security of medical data must be ensured at every stage of receiving, transferring, processing and storing information to ensure the confidentiality of patient data, as well as the availability and sustainability of health services at the same time [1]. Based on this, manufacturers of medical systems, as well as organizations that organize support, need to implement measures to ensure the necessary level of protection against cyber threats, to increase the level of safety of patients and the infrastructure of the medical institution as a whole. The following consider prevalent directions of securing medical data.

1. Ensuring incoming data conformance with DICOM standards. This should be implemented on the server-side, the DICOM server or filtering component running as a front end for the DICOM server. The datum of interest is IOD (Information Objects

- Definition). An idea to propose a formal language to express IODs is not new [2]. Modern software like dcm4che [3] supports this validation.
2. Watermarking medical images. The method proposed in [4], based on a reversible watermarking technique, provides authentication and self-correction by dividing an image into two regions: Region of Interest (ROI) and Region of Noninterest (RONI). Then, the ROI is embedded into the RONI so that any change of the image may be detected and can be self-restored back to the original image by extracting the ROI from the RONI. The work [5] proposes a security technique with patient authentication support, information confidentiality and integrity based on reversible watermarks. To provide integrity checking an MD5 (Message Digest 5) hash of the image is computed. Reversibility is achieved with compressed an R-S-Vector determined from the image (consists of bits indicating regular (1) or singular (0) state of a group of pixels). A watermark providing confidentiality and authentication services is constructed by aggregating the compressed R-S-Vector, the hash value and the patient ID. It is encrypted with AES (Advanced Encryption Standard) and embedded into medical images.
  3. Encryption of DICOM files. The work in [6] proposes the following algorithm for providing confidentiality, integrity and authenticity of the header and pixel data of DICOM images: an encryption and signature creation procedure, and a decryption and signature verification procedure. Singla and Singh [7] developed a framework proposing two different approaches to ensure cloud data security: the Extensible Authentication Protocol for authentication, and the Rijndael Encryption Algorithm used to encrypt sensitive data. Dorgham et al. [8] proposed a framework to secure transfer and storage of medical images on the cloud by using hybrid (a combination of symmetric and asymmetric) encryption algorithms. Their scheme consists of separate stages of hashing the DICOM header with SHA-3 (Secure Hash Algorithm 3) and encrypting pixel data with the result of the previous stage using an XTEA (eXtended Tiny Encryption Algorithm) algorithm.
  4. Application of artificial intelligence (AI) machinery to detect malicious tampering of medical data. Mirsky et al. [9] proposed an attack on the PACS network allowing changes in sensitive 3D imagery files using deep learning. They proposed a GAN (generative adversarial network) model helping to add or remove patterns related to lung cancer when editing 3D CT scans. The authors also mentioned use of digital signatures [10,11] as the best way to resist such attacks. They also noted that such a feature is implemented in actual PACS systems, but often is not configured in the proper way. Therefore, the supervised domain generic approach [12] is considered more robust in tampering detection.
  5. Application of User Behavior Analytics techniques [13] in Security Incident and Event Management systems. These systems contain all the data necessary to construct machine learning models, allowing discovery of patterns of abnormal activity in a dense event log, which often is considered as a big data stream [14–17]. Lee et al. [18] proposed methods to analyze security events, learn normal and threat patterns, construct event profiles and use common classification methods like Support Vector Machine, Random Forest and Naive Bayes, and proposed artificial neural network models to perform detection of malicious behavior in SIEM environments. The authors obtained accuracy near to 95% with an NSLKDD dataset [19] and 98–99% with a CICIDS2017 dataset [20]. Logical models are also applicable to model user behavior. Corapi et al. [21] proposed a model based on nonmonotonic learning allowing performance of revision of user behavior rules. They use Inductive Logic Programming machinery to train the model from examples.

Since any software may have vulnerabilities, the problem of authenticity of medical data arises, i.e., whether the file in the DICOM archive is legally added and is not fabricated. All the techniques mentioned above are necessary to ensure security of the data transferred to and stored in the archive. We propose architecture of a versatile and extensible information system designed to work in a SIEM cluster with traffic interception, analytics facilities

and alert management. The main architectural purpose is to allow integration of present and future state-of-the-art model-driven approaches of the medical data and user behavior validation mentioned in paragraphs 4 and 5 in the above.

Since human life is of the highest value, all measures to ensure data security in medical information systems are aimed at ensuring the safety of life and health of patients. A dangerous situation can arise due to the actions of an intruder gaining access to the institution's network by exploiting vulnerabilities in the external perimeter, and/or overly permissive access policies within the local computer network. An attacker can not only spoil or fabricate medical images in the DICOM archive, but also harm the patient and the institution by taking over the MIS user accounts (doctors' credentials are of particular interest) and adjusting the patient's course of treatment at his own discretion. Two measures under consideration will help protect patients and doctors from misdiagnoses and the consequences of inappropriate treatment. (1) Data protection in DICOM archives eliminating influence of damaged or fake medical images on decisions made by the diagnostician. (2) Analysis of user activity making it possible to reveal the extraordinary and dangerous behavior of MIS users associated with destructive interference in the activities of doctors. Protection of a medical institution's network implies both basic perimeter protection using firewalls and access rules implemented in network equipment, as well as the DICOM archive and application server protection components with analysis of user activity, as proposed in this paper. Thus, the protection of technical objects of the infrastructure of a medical institution ensures the protection of the life and health of patients, as well as the professional reputation of specialists in particular, and the institution as a whole.

2. Materials and Methods

This section is organized as follows. First, the formalization of the problem of data legality is presented, where three aspects are discussed: correctness of periodicity of the incoming data stream, correctness of image data (time series) for a considered modality, and legality of users' behavior. Second, we describe architectural features of subsystems intended for offline analysis of DICOM files and user activity.

2.1. Formalization of the Problem

2.1.1. Periodicity Correctness Model

Consider data series with periodically posted samples into an archive: they may be generated not only by immovable equipment, but also wearable devices, like biomedical electrocardiogram sensors. Hardware provides data on demand of software, which usually performs periodic requests and sends data to the cloud [22–25]. If periodicity of the incoming data stream is violated, it is highly likely to have an attack, if a simple loss of network connection is not the case. The following describes the proposed model of periodicity correctness for incoming data streams. It allows some variance of the periodicity to handle rare network connection issues.

Let $X = \{X_1, X_2, \dots, X_1^r, X_2^r, \dots, X_n^r, \dots, X_1^d, X_2^d, \dots, X_m^d, \dots\}$ be the time series of considered modality. It is divided into two parts: reference part (samples denoted as $X_\bullet^r$) and a part for analysis (samples denoted as $X_\bullet^d$).

Let $time(X_\bullet^r)$ be the function returning timestamp of the sample $X_\bullet^r$ passed as the argument.

Let $t_i^r = time(X_{i+1}^r) - time(X_i^r), i = 1, \dots, n - 1$ be the time interval of two samples in the reference part.

Then average time interval for the reference part $\frac{1}{n-1} \sum_{i=1}^{n-1} t_i^r$ must belong to $[T^r - \Delta^r; T^r + \Delta^r]$, where T^r is the expected period, Δ^r is allowed variation of the expected period. Both are user-defined external parameters represented by positive datetime values.

Consider $t_j^d = time(X_{j+1}^d) - time(X_j^d), j = 1, \dots, m - 1$ as the time interval of two samples in the analyzed part. The average value $\frac{1}{m-1} \sum_{j=1}^{m-1} t_j^d$ must also belong to $[T^r - \Delta^r; T^r + \Delta^r]$

and $\forall j \in \{1, 2, \dots, m\} \left(k_l \min_{i=1,2,\dots,n} (t_i^r) \leq t_j^d \leq k_u \max_{i=1,2,\dots,n} (t_i^r) \right)$ for the period of the analyzed part to be considered as valid, where k_l and k_u are user-defined external parameters represented by positive real values.

2.1.2. Legality of Medical Data

A malicious attack may involve not only the sending of knowingly fake data, breaking the periodicity, but also damage to images already recorded in the PACS. The second aspect we take into account is legality of the medical data themselves [26]. Special methods and algorithms are developed [27–30] for different modalities to perform analysis of the files and estimate possibility of fabrication of medical data with correct timestamps. Further Section 2.2.1 presents architecture of an analyzing cluster for offline analysis of DICOM archive content using a proposed periodicity correctness model and existing model-driven methods, implementation of which can be integrated via a pluggable modules mechanism. Development of such plugins is out of the scope of the present research, since the purpose is to provide their integration within a dedicated subsystem to protect a DICOM archive.

2.1.3. Legality of User Actions

User accounts also form the attack surface. An attacker can take control of a specialist's account and carry out formally correct operations leading to a drastic change in the patient's treatment plan, which can lead to a threat to life. Special methods and algorithms are developed to perform user behavior analysis on the model-driven basis. Further Section 2.2.2 presents architecture of an analyzing cluster for offline analysis of user activity using modern methods based on machine learning, implementation of which can be integrated via a pluggable modules mechanism as well as for DICOM archive content validation. Development of such plugins is out of the scope of present research, since the purpose is to provide their integration within a dedicated subsystem to protect an application server of an MIS.

2.2. New Server Roles to Ensure Medical Data Consistency

2.2.1. Validation of DICOM Archive Integrity

The proposed network analyzer (cluster) is intended for intercepting DICOM traffic and offline analysis of medical data of observed patients. Low-level software components rely on libpcap library [31] to capture traffic from network interfaces on Linux machines. All Linux functions and libpcap capabilities are labeled as LinuxKernel components in the presented UML (Unified Modelling Language) diagrams for simplicity.

The class diagram in Figure 1 illustrates the anatomy of traffic analysis components relevant to medical data. All the methods are documented in the Tables A1–A5.

The purpose of the classes on the diagram is:

1. DICOMInterceptor class is for inspecting DICOM packets, initiating content checking and generating alerts in case of detection of suspicious DICOM files. It is the main class interacting with LinuxKernel.
2. LegalityCheckingProvider class is for configuration management and providing functionality for medical data analysis (testPatientSeries, testPatientFile methods). The patient's medical data in the DICOM archive is only analyzed if tracking is enabled for the respective patient and data series (lookOnPatientSeries method). In this case, the associated machine learning models must be initialized with a training sample of regular legal data specified by a time interval (retrainPatientSeriesModel method).
3. SeriesTester class loads and uses various plugins to check legality of data for different modalities (types of data series). It also uses the instance of PeriodAnalyzer to test periodicity of incoming data.
4. PeriodAnalyzer class is for analyzing the periodicity of data incomings in accordance with the proposed periodicity correctness model (getSuspiciousFiles method).

5. ILegalityChecker is a common interface each modality-checking plugin must implement to provide functionality for initialization and utilization of the machine learning model intended for data legality checking (trainModel and testSeries methods).
6. MRChecker, RGChecker, ECGChecker are MR (Magnetic resonance imaging), RG (Radiographic imaging), ECG (Electrocardiography) modality sample checking plugins implementing ILegalityChecker interface, respectively.

Figure 1. Class diagram of the component for medical data analysis.

All classes, except for validation classes, for different modalities (MRChecker, RGChecker, ECGChecker) are implemented in the DICOMAnalyzer component. Validation classes for different modalities are implemented in separate components. The component diagram in Figure 2 illustrates the relationship of components in the system and other software with which it interacts, for example, dcm4che, which provides the functionality of a DICOM archive. The source of data for the latter is various medical hardware and software systems and wearable devices (the designation of their firmware appears in the component diagram as MREquipmentFirmware, RGEquipmentFirmware, ECGEquipmentFirmware, WearableEquipmentFirmware). The activity of the attacker's software can also affect the contents of the archive; it is denoted on the component diagram as AttackersSoftware, which also uses DCM4CHE like medical equipment.

Figure 2. Component diagram of the subsystem for medical data analysis.

DICOMAnalyzer, MRAnalyzer, ECGAnalyzer, RGAAnalyzer (and other validation components for different modalities) are packaged as separate artifacts (with the same names, Figure 3). They should all be deployed to the LegalityAnalysisBackend node. These nodes should be clustered for high availability. Inbound traffic destined for PACS is monitored at the TCPInterceptorFrontend node and routed to LegalityAnalysisBackend nodes for analysis with round-robin balancing. dcm4che, which provides the DICOM archive functionality, and PostgreSQL, which provides the database for it, are deployed on separate DICOMArchive and DICOMArchiveDatabase nodes, respectively.

Figure 3. Deployment diagram of the subsystem for medical data analysis.

The activity diagram in Figure 4 illustrates the process of inspecting and analyzing DICOM packets in a system on an analytic cluster. The process shown in the diagram assumes that the model for observing the series has already been trained (training was initiated manually or automatically using planning tools). Only valid DICOM packets can be parsed. When a packet arrives at the Linux kernel on a TCPInterceptorFrontend node, it is redirected for analysis to one of the LegalityAnalysisBackend nodes where it is checked and the timestamp value is retrieved from it (DICOMAnalyzer component). If the package being checked is a valid DICOM package, the parsing procedure runs asynchronously. It waits until the DICOM file appears in the DICOM archive, and then begins analysis for correctness of periodicity and content using the PeriodAnalyzer and [Modality]Checker instances. The getCheckerForSeries method of the SeriesTester class decides which plugin to select for content analysis. When suspicions arise, the postAlert method is called, which can provide the required functionality for alerting about threats and additional event logging. In all cases, network activity is logged (passThroughAndLog). A more detailed description of the interaction of components is illustrated in the sequence diagram in the Figure A1.

2.2.2. Validation of User Behavior

The network analyzer (cluster) is designed to intercept HTTP (Hypertext Transfer Protocol) traffic and autonomously analyze incoming events generated by specialists interacting with the interfaces of the medical information system. Examples of interfaces are a radiologist's personal account, an ECG specialist's personal account and an anesthesiologist's personal account. Low-level software components rely on mitmproxy (man-in-the-middle proxy) [32] functionality to capture traffic destined for the MIS web server. All Linux functions are labeled as a LinuxKernel component in the presented UML diagrams for simplicity.

The class diagram in Figure 5 illustrates the anatomy of the user behavior analysis components. All the methods are documented in Tables A6–A10.

The purpose of the classes on the diagram is:

1. HTTPInterceptor class is designed to parse HTTP packets, run content inspection and generate alerts in case of detection of dangerous user actions. This is the main class that interacts with mitmproxy.
2. UserActivityJournal class is designed to manage user activity log and provide data to analytic components.
3. IUserActivityJournal is an interface for working with the user activity log.
4. AuthenticityCheckingProvider class is designed to manage configuration and provide compliance checking functionality for the User Behavior Analysis Component (HTTPAnalyzer). User activity data is only analyzed if tracking is enabled for the respective user (trackUser method). In this case, the associated machine learning models must be initialized with a training sample of regular legal data specified by a time interval (retrainUserAuthModel method).
5. UserActivityTester class loads and uses various plugins to authenticate user activity to various medical professionals.
6. IAuthenticityModel is a common interface that every behavior validation plugin for any medical specialization must implement to provide functionality for initialization and utilization of machine learning model intended for data legality checking (trainModel, refineModel and testSeries methods).
7. MRSpecialistModel, CTSpecialistModel, AnesthetistModel are MR, CT (Computed tomography imaging), anesthesiologist specialist behavior validator plugins that implement the IAuthenticityModel interface, respectively.

All classes, except for the behavior testing classes for various medical specializations (MRSpecialistModel, CTSpecialistModel, AnesthetistModel), are implemented in the HTTPAnalyzer component. Behavior validation classes for different medical specializations are implemented in separate components. The component diagram in Figure 6

illustrates the relationship between the components in the system and other software with which it interacts. For example, an application server that provides the functionality of the interfaces of specialists' personal accounts. The source of events for the latter is various client devices of specialists with a web browser. The activity of the attacker's software can also affect the state of the application server; it is denoted on the component diagram as AttackerHTTPClient, which also uses the web server like specialists' browsers.

Figure 4. Analysis of incoming Digital Imaging and Communications in Medicine (DICOM) files.

Figure 5. Class diagram of the component for user behavior analysis.

HTTPAnalyzer, MRSpecialistValidator, CTSpecialistValidator, AnesthetistValidator (and other user behavior validation components for various medical specializations) are packaged as separate artifacts (with corresponding names, Figure 7). They should all be deployed to the BehaviorAnalysisBackend node. These nodes should be clustered for high availability. Inbound traffic destined for the application server is intercepted at the HTTPInterceptorFrontend node and routed to BehaviorAnalysisBackend nodes for analysis with round-robin balancing. User activity log functionality can be provided by the rsyslog component, which is deployed to a separate JournalBackend node. An application server that provides the functionality of specialists' personal accounts should be deployed on a separate WebServer node.

The activity diagram in Figure 8 illustrates the process of inspecting and analyzing HTTP packets in a system on an analytic cluster. The process shown in the diagram assumes that the behavioral models of the specialists have already been trained (training was initiated manually or automatically using the planning tools). Only valid HTTP packets carrying specialist activity data can be parsed. When a packet arrives at the mitmproxy proxy on the HTTPInterceptorFrontend node, it is redirected for analysis to one of the

BehaviorAnalysisBackend nodes, where it is checked, and the timestamp value is retrieved from it (HTTPAnalyzer component). If the package being checked is a valid HTTP package associated with a specialist activity, the following happens: the user ID is found out and, if the user is being tracked, the event is logged, and the new incoming events counter is incremented. If the minimum threshold for the number of new incoming events is reached, the procedure for analyzing them starts asynchronously. The time interval for collecting events is passed to it (eventsCollectingStartTime defines the beginning, timestamp of the last incoming event defines the end). An analysis of the authenticity of user activity using the ModalitySpecialistModel begins, the new incoming event counter is reset to zero, and the current time is captured as a new point to start collecting events from. The getCheckerForUser method of the UserActivityTester class decides which plugin to select for events analysis. When suspicions arise, the postAlert method is called, which can provide the required functionality for alerting about threats and additional event logging. In all cases, network activity is logged (passThroughAndLog). A more detailed description of the interaction of components is illustrated in the sequence diagram in Figure A2.

2.2.3. Place of Server Roles in Network Infrastructure

Figure 9 illustrates the place of network analyzers (clusters) in a typical hospital network infrastructure.

Figure 6. Component diagram of the subsystem for analysis of users' behavior.

Figure 7. Deployment diagram of the subsystem for analysis of users' behavior.

New server roles TCP Interceptor Frontend, Legality Analysis Backend, together with the existing DICOM Archive and DICOM Archive Database, form a PACS cluster for which it is advisable to allocate a separate VLAN1 (Virtual Local Area Network). Traffic that was previously directed to the DICOM Archive must be directed to the TCP Interceptor Frontend. After passing through the Legality Analysis Backend, it is directed to the DICOM Archive automatically: this is the default behavior. The latter aspect is regulated by a separate option to allow alternative analytic cluster deployment. Traffic directed directly to the DICOM Archive can be duplicated on the TCP Interceptor Frontend using port mirroring on the managed switch. In this case, routing packets from the Legality Analysis Backend to the DICOM Archive would be redundant and should, therefore, be disabled.

New server roles HTTP Interceptor Frontend, Behavior Analysis Backend and Journal Backend, together with the existing Web Server, form a Workplaces Servers Cluster, for which it is advisable to allocate a separate VLAN2. Traffic that was previously directed to the Web Server must be directed to the HTTP Interceptor Frontend. After passing through the Behavior Analysis Backend, it is directed to the Web Server automatically: this is the default behavior. The latter aspect is regulated by a separate option to allow an alternative analytic cluster deployment. Traffic directed directly to the Web Server can be duplicated to the HTTP Interceptor Frontend using port mirroring on the managed switch. In this case, routing packets from the Behavior Analysis Backend to the Web Server would be redundant and should, therefore, be disabled.

Traffic routing rules, including between VLANs, should be developed taking into account the specifics of the location of specialists' workplaces, and should not be overly permissive in order to reduce the network attack surface.

The DICOM archive, application servers, and the proposed components of their protection, form a centralized system intended for deployment within the local computer network of a medical institution. The databases required for the functioning of the protected components (DICOM Archive Database and other databases of the MIS application servers), as well as the user activity log (Journal Backend), cannot be moved outside the LAN perimeter, since they require connections with low time delays. They contain all the medical data that comes to the institution and are generated within it, being the subject of interest of this work. The system has decentralized analytical cluster subsystems to provide horizontal scalability and improve fault tolerance.

Figure 8. Analysis of user activity.

The initial start-up of system functions requires the participation of a medical specialist. The patient's medical data in the DICOM archive, as well as MIS user activity data, are only analyzed if tracking is enabled for the respective patient or user and the associated machine learning models are trained with an appropriate sample specified by a specialist with a time interval. After this, the system's behavior boils down to continuous automatic analysis of incoming big data streams and issuing warnings in case of detection of unwanted activities, e.g., suspicious DICOM files or dangerous user actions.

Figure 9. Network infrastructure with traffic analysis.

3. Discussion

Commercial SIEM solutions are common practice for healthcare facilities, as technical support is often critical. One of the leading long-standing solutions on the market is IBM QRadar [33] which embodies many of the problems raised in this work, such as allowing IBM technology partners to develop extensions, called QRadar Applications [34], including the native IBM Qradar User Behavior Analytics application [35] and also having mechanisms for maintaining high availability. Unfortunately, we could not find extensions for DICOM traffic mining. Among the undeniable architectural advantages of Qradar, it is worth noting a cluster of DataNode components, each node in which adds space and computing resources [36]. High availability relies on the presence of primary and secondary analysis hosts, with the secondary role being a standby role that is constantly synchronized with the primary.

Another common solution from AT&T Cybersecurity, according to a Gartner study [37], is USM (Unified Security Management) Appliance [38], which also contains UBA modules but does not allow the development of third-party extensions; only native plugins that parse events using regular expressions are allowed [39]. The HA support mechanism relies on a passive copy of the installation (slave) [40].

The architecture proposed does not imply limitation on the number of analytical backends in the cluster. Therefore, from the point of view of horizontal scalability, it can be considered more reliable than the considered solutions. However, intercepting front-end servers can be considered a single point of failure. To improve reliability, they can have a backup “twin”. There is no need to cluster them with load balancing over TCP connections, since the primary traffic processing is lightweight and does not use local file systems to store any intermediate data.

4. Conclusions

The architecture of analytical clusters is used to improve functionality of SIEM environments deployed in hospitals. It is designed specifically to allow integration of validation of medical data itself and user behavior analytics to validate medical specialists’ activity. The practical impact of software implementation and deployment improves the security

of sensitive medical imagery. Since there are no requirements for plugins, except of implementation of one common interface which provides ML-model management, there are no additional technical efforts when implementing any model-driven approach for data validation. This allows programmers to extend the system with new analyzers when new effective methods appear.

Software implementation of the proposed architectural principles can form the basis of a framework designed to collect and analyze medical data for its authenticity, as well as user behavior data for its legality. The availability of such a toolkit will allow researchers working in any of these areas to simplify the tasks of collecting data and conducting experiments when debugging machine learning models.

The development of fault-tolerant event stores is out of the scope of this article, but is one of the subjects for further research, namely optimizing the Apache Hadoop stack [41] to build an event store ready for big data analysis. The Apache Spark Mllib [42] implementations of the many machine learning algorithms provide a suitable integrated solution for use in analytic plugins. API development, implementation of thin wrappers for such use, as well as analysis and optimization of performance when handling events, is the subject of further work.

Author Contributions: Conceptualization, S.M.; methodology, A.L.; validation, S.M.; formal analysis, S.M.; investigation, A.L. and S.M. writing—original draft preparation, A.L.; writing—review and editing, S.M.; supervision, S.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Classes for Validation of DICOM Archive Integrity

Table A1. Members of DICOMInterceptor class.

Member Name	Member Purpose
inspectPacket	1. Extracts timestamp from the data packet. 2. If data is valid DICOM file with timestamp, run the analyzing task asynchronously.
passThroughAndLog	Passes the packet through and logs this event.
extractTimestamp	Extracts timestamp from DICOM file.
postAlert	Posts alert for specified DICOM file.
analysisTaskProduce	1. Waits for the file with the specified timestamp appears in the DICOM archive. 2. Runs the analysis of this file. 3. If there are violations, posts alerts.

Table A2. Members of LegalityCheckingProvider class.

Member Name	Member Purpose
retrainPatientSeriesModel	Retrain ML model for specified series with specified data interval for training.
testPatientSeries	Returns hashmap with samples IDs as keys and their legality as <boolean, boolean> values. The first boolean in the tuple is period legality, the second boolean in the tuple is content legality.
testPatientFile	Returns period and content legality of the single specified patient file.

Table A2. Cont.

Member Name	Member Purpose
getSeriesOfPatient	Return observed series IDs for the patient.
releasePatientSeries	Remove the patient's series from the observation list.
lookOnPatientSeries	Add a series of a patient to the observation list.
observations	Hashmap with patient IDs as keys and lists of series IDs as values.

Table A3. Members of SeriesTester class.

Member Name	Member Purpose
testSeries	Returns hashmap with samples IDs as keys and their legality as <boolean, boolean> values. The first boolean in the tuple is period legality, the second boolean in the tuple is content legality.
getCheckerForSeries	Get the entry point in the appropriate plugin for specified data series.
periodAnalyzer	An instance of PeriodAnalyzer.

Table A4. Members of PeriodAnalyzer class.

Member Name	Member Purpose
setEthalonPeriod	Sets the value of the period to check conformance of the time series with it.
getSuspiciousFiles	Returns hashmap with samples IDs as keys and their legality as boolean values.

Table A5. Methods of ILegalityChecker interface.

Method Name	Method Purpose
trainModel	Train ML model for specified series with specified data interval for training.
testSeries	Returns hashmap with samples IDs as keys and their legality as boolean values.

Figure A1. Analysis of incoming DICOM files (sequence diagram).

Appendix B. Classes for Validation of User Behavior

Table A6. Members of HTTPInterceptor class.

Member Name	Member Purpose
minEventsThreshold	A minimum threshold of new events count to start analysis.
eventsCollectingStartTime	A timestamp when collecting new events has started.
newEventsCounter	A counter of new incoming user activity events to be analyzed further.
extractTimestamp	Extracts timestamp from HTTP headers.
incNewEventsCounter	Increments the counter of the new user activity events.
resetNewEventsCounter	Resets the counter of new user activity events to zero. Sets events collecting start time to current time.
getNewEventsCounter	Returns value of the counter of new user activity events.
postAlert	Posts alert for specified user activity time interval.
analysisTaskProcedure	1. Runs the analysis of new events. 2. If there are suspicions, posts alerts.
inspectPacket	1. Extracts timestamp from the data packet. 2. If data is valid HTTP with user action, increment counter of new events, and if it reaches minEventsThresholds, runs the analyzing task asynchronously.
passThroughAndLog	Passes the packet through and logs this event.

Table A7. Members of AuthenticityCheckingProvider class.

Member Name	Member Purpose
trackedUsers	List of IDs of tracked users.
trackUser	Add user identified by userID to tracking list. True is returned if the appropriate behavior model plugin is accessible and entry is not added yet.
releaseUser	Do not track user identified with userID anymore. True is returned if entry exists in the tracking list.
testUserActivitySeries	Returns hashmap with event timestamps as keys and their authenticity estimation as boolean values. ML model source is defined automatically by userID.
retrainUserAuthModel	Retrain ML model for specified user with specified data interval for training.
getUserIDByActivityEvent	Returns user ID by its activity event.

Table A8. Members of UserActivityTester class.

Member Name	Member Purpose
getCheckerForUser	Get the entry point in the appropriate plugin for specified user behavior model.
testSeries	Returns hashmap with event timestamps as keys and their authenticity estimation as boolean values. ML model source is defined by user identified by modelUserID, test data is defined by activity of the user identified by userID. If users identified by modelUserID and userID have incompatible ML model types, an exception is raised.
refineModel	Refine model for user identified by modelUserID with data associated with activity of the user identified by userID. If users identified by modelUserID and userID have incompatible ML model types, an exception is raised.

Table A9. Methods of IAuthenticityModel interface.

Method Name	Method Purpose
trainModel	Train ML model for a user specified with userID with specified data interval for training. The model is saved elsewhere to be used by testSeries and refineModel methods.
testSeries	Returns hashmap with event timestamps as keys and their estimation as boolean values. «True» stands for authentic, «False» stands for illegal. ML model used is associated with the user identified by modelUserID, data series are defined by activity of user identified by userID. Users identified by modelUserID and userID must be associated with the same ML model type.
refineModel	Refine model for user identified by modelUserID with data associated with activity of the user identified by userID. Users identified by modelUserID and userID must be associated with the same ML model type.

Table A10. Methods of IUserActivityJournal interface.

Method Name	Method Purpose
trainModel	Query events from journal for specified time interval and user identifier.
testSeries	Record event data to journal with specified timestamp and user identifier.

Figure A2. Analysis of user activity (sequence diagram).

References

- Magomedov, S.G. Security analysis of computer networks and applications of the healthcare organizations information processes. *Cloud Sci.* **2020**, *7*, 685–704.
- Hewett, A.J.; Grevenmeyer, H.; Barth, A.; Eichelberg, M.; Jensch, P.F. Conformance testing of DICOM image objects. In *Medical Imaging 1997: PACS Design and Evaluation: Engineering and Clinical Issues*; International Society for Optics and Photonics: Bellingham, WA, USA, 1997; Volume 3035, pp. 480–487.
- Open-Source Clinical Image and Object Management. Available online: <https://www.dcm4che.org/> (accessed on 13 January 2021).
- Coatrieux, G.; Montagner, J.; Huang, H.; Roux, C. Mixed reversible and RONI watermarking for medical image reliability protection. In Proceedings of the 2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society IEEE, Lyon, France, 22–26 August 2007; pp. 5653–5656.
- Abd-Eldayem, M.M. A proposed security technique based on watermarking and encryption for digital imaging and communications in medicine. *Egypt. Inform. J.* **2013**, *14*, 1–13. [\[CrossRef\]](#)
- Al-Haj, A. Providing integrity, authenticity, and confidentiality for header and pixel data of DICOM images. *J. Digit. Imaging* **2015**, *28*, 179–187. [\[CrossRef\]](#)
- Singla, S.; Singh, J. Cloud data security using authentication and encryption technique. *Glob. J. Comput. Sci. Technol.* **2013**, *13*, 2232–2235.

8. Dorgham, O.; Al-Rahamneh, B.; Almomani, A.; Khatatneh, K.F. Enhancing the security of exchanging and storing DICOM medical images on the cloud. *Int. J. Cloud Appl. Comput.* **2018**, *8*, 154–172. [CrossRef]
9. Mirsky, Y.; Mahler, T.; Shelef, I.; Elovici, Y. CT-GAN: Malicious tampering of 3D medical imagery using deep learning. In Proceedings of the 28th {USENIX} Security Symposium ({USENIX} Security 19), Santa Clara, CA, USA, 14–16 August 2019; pp. 461–478.
10. Cao, F.; Huang, H.K.; Zhou, X.Q. Medical image security in a HIPAA mandated PACS environment. *Comput. Med. Imaging Graph.* **2003**, *27*, 185–196. [CrossRef]
11. Digital Imaging and Communications in Medicine (DICOM). Available online: <https://www.dicomstandard.org/> (accessed on 22 January 2020).
12. Cozzolino, D.; Thies, J.; Rössler, A.; Riess, C.; Nießner, M.; Verdoliva, L. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv Preprint* **2018**, arXiv:1812.02510.
13. Csaba, K.; Péter, H.B. Analysis of Cyberattack Patterns by User Behavior Analytics. *AARMS–Acad. Appl. Res. Mil. Sci.* **2018**, *17*, 101–114.
14. Veeramachaneni, K.; Arnaldo, I.; Korrapati, V.; Bassias, C.; Li, K. AI²: Training a big data machine to defend. In Proceedings of the 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), New York, NY, USA, 9–10 April 2016; pp. 49–54.
15. Magomedov, S.; Ilin, D.; Silaeva, A.; Nikulchev, E. Dataset of User Reactions When Filling Out Web Questionnaires. *Data* **2020**, *5*, 108. [CrossRef]
16. Nikulchev, E.; Ilin, D.; Silaeva, A.; Kolyasnikov, P.; Belov, V.; Runtov, A.; Pushkin, P.; Laptev, N.; Alexeenko, A.; Magomedov, S. Digital Psychological Platform for Mass Web-Surveys. *Data* **2020**, *5*, 95. [CrossRef]
17. Magomedov, S.G.; Kolyasnikov, P.V.; Nikulchev, E.V. Development of technology for controlling access to digital portals and platforms based on estimates of user reaction time built into the interface. *Russ. Technol. J.* **2020**, *8*, 34–46. (In Russian) [CrossRef]
18. Lee, J.; Kim, J.; Kim, I.; Han, K. Cyber Threat Detection Based on Artificial Neural Networks Using Event Profiles. *IEEE Access* **2019**, *7*, 165607–165626. [CrossRef]
19. Tavallae, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
20. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSP* **2018**, *1*, 108–116.
21. Corapi, D.; Ray, O.; Russo, A.; Bandara, A.; Lupu, E. Learning rules from user behaviour. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*; Springer: Boston, MA, USA, 2009; pp. 459–468.
22. Dehlinger, J.; Dixon, J. Mobile application software engineering: Challenges and research directions. *Workshop Mob. Softw. Eng.* **2011**, *2*, 29–32.
23. Kassinen, O.; Harjula, E.; Koskela, T.; Ylianttila, M. Guidelines for the implementation of cross-platform mobile middleware. *Int. J. Softw. Eng. Appl.* **2010**, *4*, 43–58.
24. Petrov, A.V.; Bolshakov, O.S.; Lebedev, A.S.; Golubeva, N.E. Application template method to increase mobility of distributed systems for collecting and relaying information from biomedical sensors. *J. Radio Electron.* **2013**, *5*, 7. Available online: <http://jre.cplire.ru/iso/may13/5/text.pdf> (accessed on 22 January 2020). (In Russian).
25. Lebedev, A.S.; Bolshakov, O.S.; Petrov, A.V. Designing distributed retransmission system with the mobile clients based on cross-platform software development methods. *Curr. Probl. Sci. Educ.* **2014**, *1*, 227. Available online: <https://science-engineering.ru/pdf/2014/1/352.pdf> (accessed on 22 January 2020). (In Russian).
26. Karpov, O.E.; Akatkin, Y.M.; Konyavsky, V.A.; Shishkanov, D.V.; Yasinovskaya, E.D. Digital health in a digital society. *Ecosyst. Clust.* **2017**, *220*, 48.
27. Komisaruk, O.V.; Nikulchev, E.V.; Malykh, S.B. Neural network model for artifacts marking in EEG signals. *Cloud Sci.* **2020**, *7*, 631–654. Available online: https://www.researchgate.net/publication/341882969_Razrabotka_nejrosetevoj_modeli_vyavleniya_artefaktov_v_elektroencefalogramme_mozga (accessed on 22 January 2020). (In Russian).
28. Benssalah, M.; Rhaskali, Y. A Secure DICOM Image Encryption Scheme Based on ECC, Linear Cryptography and Chaos. In Proceedings of the 2020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), El-Oued, Algeria, 16–17 March 2020; pp. 131–136.
29. Mortajez, S.; Tahmasbi, M.; Zarei, J.; Jamshidnezhad, A. A novel chaotic encryption scheme based on efficient secret keys and confusion technique for confidential of DICOM images. *Inform. Med. Unlocked* **2020**, *20*, 100396. [CrossRef]
30. Shini, S.G.; Thomas, T.; Chithranjan, K. Cloud based medical image exchange-security challenges. *Procedia Eng.* **2012**, *38*, 3454–3461. [CrossRef]
31. TCPDUMP/LIBPCAP Public Repository. Available online: <https://www.tcpdump.org/> (accessed on 22 January 2020).
32. Mitmproxy—An Interactive HTTPS Proxy. Available online: <https://mitmproxy.org/> (accessed on 22 January 2020).
33. IBM QRadar SIEM – Overview. Available online: <https://www.ibm.com/products/qradar-siem> (accessed on 22 January 2020).
34. IBM Technology Partners. Available online: <https://www.ibm.com/support/pages/technology-partners> (accessed on 22 January 2020).
35. IBM QRadar User Behavior Analytics (UBA) App: User Guide. Available online: https://www.ibm.com/support/knowledgecenter/SS42VS_SHR/com.ibm.UBAapp.doc/b_Qapps_UBA.pdf (accessed on 22 January 2020).

36. IBM QRadar: Architecture and Deployment Guide. Available online: https://www.ibm.com/support/knowledgecenter/en/SS42VS_7.3.3/com.ibm.qradar.doc/b_siem_deployment.pdf (accessed on 22 January 2020).
37. Gartner Magic Quadrant for Security Information and Event Management. Available online: <https://www.gartner.com/en/documents/3981040/magic-quadrant-for-security-information-and-event-manage> (accessed on 22 January 2020).
38. About USM Appliance. Available online: <https://cybersecurity.att.com/documentation/usm-appliance/system-overview/about-usm-solution.htm> (accessed on 22 January 2020).
39. USM Appliance. Develop New Plugins from Scratch. Available online: <https://cybersecurity.att.com/documentation/usm-appliance/plugin-management/developing-new-plugins.htm> (accessed on 22 January 2020).
40. Configuring High Availability in USM Appliance Enterprise Systems. Available online: <https://cybersecurity.att.com/documentation/usm-appliance/configuring-ha/deploying-ha-in-usm-enterprise-prods.htm> (accessed on 22 January 2020).
41. Apache Hadoop. Available online: <https://hadoop.apache.org/> (accessed on 22 January 2020).
42. MLlib. Apache Spark. Available online: <https://spark.apache.org/mllib/> (accessed on 22 January 2020).