A Key Management Framework to Secure IoMT-enabled Healthcare Systems

Marcus de Ree*, Damian Vizár†, Georgios Mantas*‡, Joaquim Bastos*,
Corinne Kassapoglou-Faist† and Jonathan Rodriguez*§

*Mobile Systems Group, Instituto de Telecomunicações, Aveiro, Portugal
†CSEM S.A., Neuchâtel, Switzerland
‡Faculty of Engineering and Science, University of Greenwich, Chatham Maritime, UK
§Faculty of Computing, Engineering and Science, University of South Wales, Pontypridd, UK
Email: mderee@av.it.pt, damian.vizAr@csem.ch, gimantas@av.it.pt, jbastos@av.it.pt, jonathan@av.it.pt

***Abstract*—The transformation of the healthcare sector through the adoption of the Internet of Medical Things (IoMT) provides major benefits, including the ability to provide efficient and timely medical support based on accurate continuous monitoring data. However, the necessity to collect, store, and process private medical data in order to provide a patient with these healthcare services may clash with regulations such as the General Data Protection Regulation (GDPR). In this article, we introduce a complete key management framework for an IoMT patient monitoring system. The key management framework is composed of a platform key management layer which establishes ad-hoc, point-to-point secure channels between devices in the IoMT system, and of a data key management layer which provisions keys for end-to-end encryption of patient data. The cornerstone of the design is that it empowers the patient to enforce their own privacy rights by making them the legal owner of their own private medical data and that interested parties must be granted consent in order to access this data. To the authors' knowledge, this is the first time that consent granting of GDPR is hardwired into technology.**

***Index Terms*—Internet of Medical Things, Key Management, Patient Monitoring, Privacy, Security**

I. Introduction

The Internet of Things (IoT) refers to a wide range of interconnected objects and devices that harvest information from the environment through sensors, analyze it and act back on the physical world through actuators [1], [2]. In the healthcare sector, IoT devices, also known as the Internet of Medical Things (IoMT), may support core functions of healthcare and health-related services [3], [4]. However, the lack of a standardized and lightweight security framework for IoMT applications can pose various security and privacy threats to the medical data, its reliability and timely availability [5]–[7]. In the case of emergency, it is important to have direct access to a patient's personal medical data (even though this may include private data that is not applicable to the emergency) while preserving a patient's privacy rights as mandated by regulations such as the General Data Protection Regulation (GDPR) [8] in all other scenarios (i.e., non-emergency situations) [9]. It is vital for a key management framework to find the right balance between security and the protection of personal data, which remains an open issue for medical networks [3].

End-to-end security in IoT networks has been an active area of research. However, common design flaws include the expectation of routers to re-encrypt the sensed data [10] or the utilization of static pre-shared keys [11] which are considered bad practice for real-world implementations. Furthermore, end-to-end security schemes [12] only focused on the secure data transfer from sensing device to storage platform without incorporating data sharing or concent management capabilities that allow the provider of private sensing data to enforce its privacy rights.

In this article, we describe a novel key management framework that is primarily designed for continuous patient monitoring applications although it could be applied to medical applications in general. The philosophy and main security goals behind the design of our key management framework are as follows:

1) **Patient health.** The main requirement for this objective is the timely availability of good quality monitoring data in the hospital cloud for automatic evaluation, diagnostics, and medical alerts, as well as for direct browsing by healthcare professionals (in case of emergency). The access to the data in the cloud may be necessary by third parties (e.g., an independent healthcare provider).
2) **Patient-centric security.** As the legal owner, a patient has the ultimate right to their sensitive monitoring data. Yet, the patient typically has little to no means to enforce these rights. This key management framework empowers the patients and making them the legal owner of their medical data (i.e., the patient is the owner of the cryptographic keying material, used to securely store and required to access their medical data). The process of requesting consent from a patient to access certain personal monitoring data and the technical means to access it is integrated into a single, non-repudiable interaction. This feature gives the patient direct control over who can access their private data.
3) **End-to-end security.** Aiming at surpassing the state of the art, this key management framework ensures confidentiality and integrity of monitoring data from the point of origin (i.e., the sensing devices) to

This work is part of the Moore4Medical project funded by the ECSEL Joint Undertaking under grant number H2020-ECSEL-2019-IA-876190.

the point of processing (i.e., the healthcare cloud storage).

This article is outlined as follows. In section II, we describe two patient monitoring use-cases and present the network model for which the novel key management framework is designed. In section III, we describe the platform key management which constitutes a lightweight authenticated key exchange procedure, based on the Kerberos system, to enable secure data transmission. In section IV, we describe the data key management for end-to-end security wrapped aroud patient's privacy. Finally, section V concludes this article with a summary of its main contributions.

II. Scenario Architecture

A. Use Cases

This key management framework is primarily designed for secure continuous patient monitoring applications in IoMT networks [13]. There are various reasons for healthcare professionals to monitor their patients [14]. One use-case relates to patients that undergo cardiac surgery. They must be monitored to detect possible atrial fibrillation events which may lead to long-term or permanent post-surgery complications (such as brain strokes). With the assistance of IoMT devices, the patient can be continuously monitored from the hospital as well as from the patient's home. Another use-case for continuous patient monitoring is related to sleep disorders, such as obstructive sleep apnea, which cause complications such as cardiovascular diseases. With the assistance of IoMT devices, the amount of physical activity and overall sleep quality of a patient can be monitored and analyzed. This data allows health care personnel to provide recommendations to the patient and improve their sleep routines.

In both cases, it is critical that the monitoring system is (i) easy to use, such that patients and health care personnel can make use of the system as intended; (ii) reliable (i.e., no crashes and no loss of data), such that every atrial fibrillation event or physical sleep activity can be captured, potentially alerting health care personnel whenever necessary; and (iii) secure, such that the monitoring data can be securely transmitted to a cloud storage where it can be securely stored for a long period of time.

B. Network Model

Based on the previously described use-cases, we propose the network model as depicted in Fig. 1. In this model we consider three parties: the patient, a healthcare facility (e.g., hospital), and interested third parties (e.g., partnered universities, research institutions).

The patient is equipped with a patient personal device (PPD) that they own, or at least physically controls, and trust to ensure the security of their data. This device acts as the patient's root of trust for the security of their monitoring data and serves as the patient's electronic representation in the system. As an example, this can be a dedicated, low-power, and highly portable device with solid physical security features (e.g., a bracelet). If a trade-off between trust and flexibility is desirable, this can be a virtual device making use of a secure element in a smartphone [15]. As will be covered in more detail in section III and IV, the trusted device is used to (i) handle patient personal authentication; (ii) manage monitoring data encryption keys, derived from a single master key belonging to the patient and shared with no one; and (iii) handle real-time electronic data processing consent requests and replies.

The sensor devices, gateway devices, and cloud storage service are assumed to be fully controlled by the healthcare facility, in particular, in terms of their (re)personalization. The variety of sensing devices, possibly based on heterogeneous technologies by different manufacturers, produce data by monitoring the patient or their immediate environment. The transmitted sensing data is received by one (or more) gateway(s) through a short-range wireless connection (e.g., Bluetooth/BLE, WiFi, ZigBee). The sensing devices and gateway(s) are within physical proximity of the patient and constitute to the monitoring platform, which may be inside the healthcare facility or installed at a patient's home. The gateway(s) forwards the sensing data to the cloud for processing and limited-duration storage. The monitoring data may be used for several purposes, including direct inspection by healthcare professionals, patient diagnosis, generation of medical alerts, and inspection by a third party (e.g., for research purposes or treatment of the same patient by another facility). However, the monitoring data is the legal property of the patient and should only be collected, analyzed, and viewed when and if the patient gives consent.

III. Platform Key Management

The purpose of the platform key management is to enable secure data transmission by establishing ad-hoc secure communications channels between the sensors, gateways, PPDs, and the healthcare cloud storage service. The establishment of point-to-point secure communications channels between devices of the monitoring platform is a pre-requisite for the data key management, covered in section IV. At the same time, such secure channels also enable a number of other advanced security functions such as intrusion detection or threshold cryptography.

A. Secure Data Transmission

Authentication and key establishment are fundamental steps in setting up secure communications channels. Authentication is concerned with knowing that the legitimate entities are communicating; key establishment is concerned with obtaining proper cryptographic keys to protect the communications, particularly to provide confidentiality and integrity of the data communicated [16]. Our network model defines a variety of communications channels that must allow the secure transmission of data, including the channel between a sensor and a nearby gateway. The sensors, resource constrained devices in terms of computational capabilities, storage capacity, and battery life would benefit from a lightweight key management solution. Therefore, the use of key management frameworks that rely on public key cryptographic (e.g., Public Key Infrastructure) are discouraged for our application.

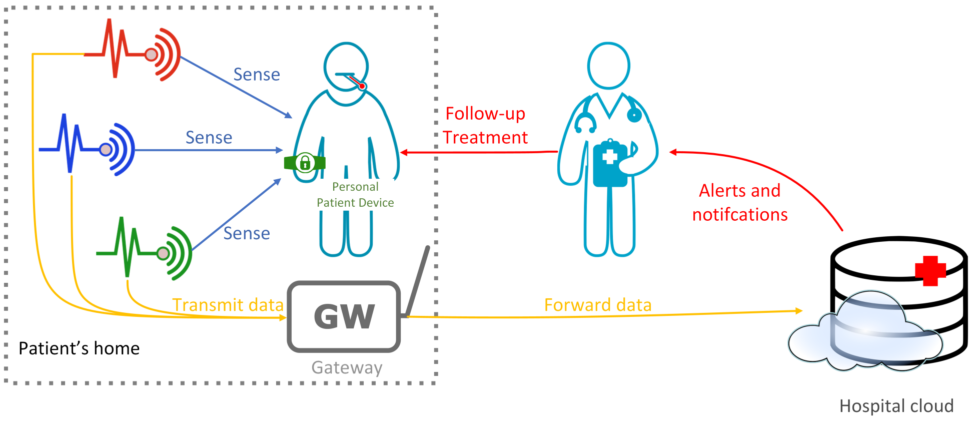

Fig. 1. The proposed network model for continuous patient monitoring applications.

For the platform key management, we specify two procedures: device bootstrapping and the secure channel establishment. These procedures specify our authenticated key exchange (AKE) that has many similarities to Kerberos [17]. Since the development of Kerberos, it has been implemented and integrated in many online services as a lightweight, trustworthy, and reliable means for establishing secure communication in unsecured networks [18]. Due to these characteristics, Kerberos has recently been mentioned as a viable solution to secure IoT systems [18]–[20]. For our healthcare application, where resource-constrained sensing devices must establish a secure channel with a nearby gateway, the sensing devices benefit from a low memory storage overhead, communication overhead, and computational overhead.

Device Bootstrapping. We consider that the device bootstrapping, also known as registration, takes place at the healthcare facility with the assistance of technical personnel. Also, we assume that the key distribution center (KDC) is a physically secured server, accessible by the healthcare facility's technical personnel for the purpose of registering the network devices. We denote the healthcare-controlled cloud storage service and the healthcare-controlled devices (e.g., sensors, gateways) as a device D with identity $ID(D)$. In the case of the sensors and gateways, we denote the identity $ID(Patient)$ for the patient identity that it is assigned to for monitoring purposes. The KDC, in possession of a master key K_{KDC}, utilizes a pseudo-random function (PRF) and a random salt to generate the long-lived symmetric key K_D for a device D as follows:

$$K_D = PRF(K_{KDC}, ID(D), salt_D) \tag{1}$$

The KDC stores the 3-tuple ($ID(D)$, $ID(Patient)$, $salt_D$) in its database for future authentication purposes. The registration of the healthcare cloud storage service is stored as ($ID(D)$, $[empty]$, $salt_D$).

As mentioned, we consider that the device bootstrapping of the PPD also takes place at the healthcare facility. Since the patient will be participating in a continuous monitoring application, it is not hard to assume that the patient has physically been present at the healthcare facility prior to the initiation of the patient monitoring. However, the exact nature of the registration procedure will depend on the specifics of the PPD. If we consider a bracelet, then the long-lived symmetric key can be installed through a short-range technology (e.g., NFC). If we consider a smartphone, then the long-lived symmetric key can be derived from the authentication code (e.g., password, biometric fingerprint) that is used to log into the appropriate application. Either way, the PPD, denoted as device D with identity $ID(D)$ and controlled by patient with patient identity $ID(Patient)$, registers at the KDC and securely obtains the long-lived symmetric key according to the structure of equation (1). The careful design for device bootstrapping is motivated by the following:

- The master key K_{KDC} and pseudo-random function PRF serve the purpose of limiting the memory storage overhead for the KDC [21]; the symmetric key K_D can be generated on-demand and therefore does not need to be stored in the database of the KDC.
- The salt serves the purpose of key updating [21].
- The incorporation of $ID(Patient)$ at the sensors, gateways, and PPDs serves authentication purposes. Namely, it is preferred that the devices assigned to a particular patient can establish a secure channel whereas devices within close proximity but assigned to different patients cannot. Thus, the PPD can only provide data encryption keys to its assigned sensors and the encrypted sensing data can only be routed to the healthcare cloud storage through the patient-assigned gateway.

Secure Channel Establishment. Once the network entities (e.g., sensors, gateways, PPD, healthcare cloud storage) are bootstrapped and share a symmetric key with the KDC, they can send requests to the KDC for the establishment of a secure channel (i.e., the establishment of a pairwise symmetric key). This secure channel establishment procedure must be initiated at the start of a patient monitoring session such that the PPD can securely transmit data encryption keys to the sensors. Since the patient initiates the monitoring session, the PPD should initiate the process. In the case of a bracelet, the process may start every time that the bracelet is turned on. In the case of a smartphone, the process may start every time that a monitoring functionality is activated on the device. This process may kick off by transmitting beacon signals for the discovery of nearby devices after which the KDC is contacted for secure channel establishment between the devices. In the description of the secure channel establishment, we denote the initiating network entity by client A and we denote client B as the network entity with whom the secure channel is established. These clients can be any combination of network entities. This procedure, as depicted in Fig. 2, is very similar to the description of secure channel establishment in Kerberos.

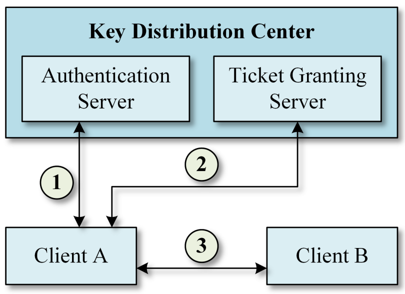

Fig. 2. The communication flow for establishing a secure channel between client A and client B

The first message exchange takes place between client A and the authentication server (AS). The client A sends an authentication request to the AS. The AS then verifies the request by looking up the registration details of client A. When authentication is successful, the AS grants client A with a ticket-granting ticket.

The second message exchange takes place between client A and the ticket granting server. The client A requests to communicate with client B and provides the ticket-granting ticket to the ticket granting server. The ticket granting server verifies the legitimacy of this request. Assuming that both clients were registered, the request will only be legitimate in the following two cases: (i) neither of the clients is the healthcare cloud storage and both client are registered as network entities that belong to the same patient, or (ii) one of the clients is the healthcare cloud storage. Therefore, if client A is a gateway assigned to a patient X and client B is a sensor assigned to a patient Y then the request will be denied. If successful, the ticket granting server starts the verification of the ticket-granting ticket. If the ticket-granting ticket is also successfully verified, it responds to client A with a symmetric key K_{AB} and a ticket that is intended for client B.

The third message exchange takes place between clients A and B. The client A sends the received ticket to client B. This ticket contains the symmetric key K_{AB}, an indicator for the lifetime that the key is valid and a timestamp of when the ticket was granted. The timestamp allows client B to verify that the ticket has recently been granted. The client B then responds back to client A to confirm that the shared symmetric key K_{AB} has been received.

B. *Distributed Kerberos*

So far, we considered the KDC as a single trusted entity who provides an authentication service and utilizes tickets to securely distribute a session key to a pair of network entities. The security of a such a key distribution protocol depends on the assumption that the AS, as part of the KDC, is trustworthy. Unfortunately, not every AS can be considered trustworthy. If the AS becomes compromised and malicious, the security of communications between the network entities cannot be guaranteed [22]. The authentication service is also a performance bottleneck because the session key distribution cannot proceed unless the identities of the network entities can be satisfactorily established. A desirable authentication service should therefore be highly available and highly secure at the same time [23].

To design a distributed Kerberos system (i.e., a distributed KDC), research efforts targeted distributed PRFs. Distributed PRFs support the splitting of the master key K_{KDC} among n servers such that at least a threshold t servers are needed to evaluate the PRF. Evaluating the PRF is done without reconstructing the key at a single location [24]. A distributed Kerberos system can therefore provide resiliency against bottlenecks and a single point of failure. The first proposal, based on the distributed evaluation of *weak* PRFs, was described in [25]. More recently, the study of key homomorphic PRFs (i.e., PRFs that are homomorphic with respect to its key) gave rise to novel constructions [26]. The most notable works on constructions for key homomorphic PRFs found in literature are by Boneh et al. [26], Banerjee et al. [27], and Kim [28], which can aid in the design of a distributed Kerberos system.

IV. Data Key Management

The objective of our data key management is to provide secret keys for end-to-end encryption of the patient's monitoring data, while letting the patient have full control over key sharing and thus, technologically supporting the GDPR. We assert that the patient data need to be secured end-to-end: encrypted at the sensor immediately after acquisition, decrypted by the cloud applications at processing time, they are not exposed anywhere in-between (in transit or in storage).

The necessity of the end-to-end approach is best illustrated by the recently surging attacks against IoT devices [29], [30]. By securing the data end-to-end (and avoiding re-encryption in several points), the attack surface for successfully modifying and/or exfiltrating the patient data is greatly reduced: on one hand, we ensure that the Internet-exposed gateway cannot leak the data, and on the other hand, the data is less exposed to the networking and computational infrastructure of the cloud. The crux of sound end-to-end security is key management, ensuring that the keys are available to sensors for encryption and to the healthcare provider cloud application(s) for decryption whenever needed, not to mention the authenticity and confidentiality requirements on the distribution. In section IV-A, we describe the design of our data key management system. In section IV-B, we show how it can be extended to server electronic data processing consent requests.

A. End-to-End Monitoring Data Key Management

A common way to design end-to-end key management is based on a centralized key server run by the hospital, which provisions the data encryption (resp. decryption) keys to both the sensors (or data sources), and to the cloud application(s) in need of data processing. The disadvantage of this approach is that the patient must fully trust the hospital with the security of all their data. Moreover, in the future, a patient is likely going to be treated and monitored by several healthcare facilities during their life, requiring them to dissipate their trust among multiple organizations. We therefore propose a paradigm shift, where the patient physically controls a personal key management device, the PPD, used for all instances of monitoring of the patient, in any facility.

Personal Key Manager. The root of trust for the patient is the PPD. The PPD securely stores a single secret key K_{master} securely generated when the PPD is personalized by the patient, potentially for the patient's lifetime. The master key is only used by the PPD to derive keys for the sensors and for the healthcare facility.

Prior to the monitoring, each sensor requests key provisioning from the PPD. The request message contains optional information I_{sensor} about the sensor, and the configuration $Conf$ of the key schedule to be used by the sensor (see below). The PPD computes a data protection key K_{data} computed as per equation (2), using a pseudorandom function PRF, where the public data key metadata string $meta_{K_{data}}$ encodes a uniformly randomly generated unique ID of the data key, $Conf$, I_{sensor}, and optional information I_{PPD} about the PPD. The PPD relies on the sensor with K_{data} and $meta_{K_{data}}$.

$$K_{data} = PRF(K_{master}, meta_{K_{data}}) \tag{2}$$

The entire interaction of the data key provisioning relies on the secure channel establishment by the platform key management, with the advantage that, once PPD is added to the domain of the healthcare provider, it can provision any number of sensors without any additional configuration.

Encryption Keys. The sensor applies a key schedule to derive data encryption keys K_{enc} from K_{data}, through a number of intermediate secret values. The intermediate secret values are organized in a tree structure rooted in K_{data} with encryption keys as leaves, where each node is derived with PRF using its parent as secret key, and a public tree node label as input. Each encryption key K_{enc} is then identified with a public metadata string $meta_{K_{enc}}$, which encodes the public node labels on the path from K_{data} to K_{enc}. This facilitates an efficient sharing of decryption keys for multiple encrypted records simultaneously. For example, considering a tree of depth two that is organized according to time, with a node per month on level one (intermediate value) and a node per week on level two (encryption key), it would be used to efficiently share all data acquired during a week by sharing an encryption key, or during a month by sharing an intermediate value with the processing application; the per-week encryption keys can then be derived locally by applying PRF with the public tree labels. In general, possession of an intermediate secret value allows all encryption keys in its subtree to be derived. Finally, the sensor also regularly derives a "refreshed" K_{data} by applying PRF with an input ensuring domain separation from the tree structure for pragmatic forward secrecy. The frequency of the refreshes, as well as the depth and shape of the tree structure are all defined in $Conf$.

Sharing of Decryption Keys. The healthcare provider collects encrypted data records augmented with $(meta_{K_{data}}, meta_{K_{enc}})$ pairs from its sensors but receives no keys. To get the decryption key for a record, the processing application needs to send the metadata pair to the PPD, which is then able to reconstruct the full sequence of PRF calls leading from K_{master} to K_{enc}, solely from the metadata. To get decryption keys to a number of records, all sharing the same $meta_{K_{data}}$, the processing application examines their $meta_{K_{enc}}$ strings, and computes $meta$ string identifying an intermediate tree node that has all the necessary decryption keys in its subtree, and send $(meta_{K_{data}}, meta)$ to the PPD. If the actually needed decryption keys are too sparse among the leaves of such a subtree, efficiency can be traded for privacy by making several requests, each targeting a smaller number of records.

Key Backup. Because K_{master} is being used for such an extensive period, it is imperative that it can be backed up. Otherwise, critical health-related data of a patient may be lost. For this purpose, the PPD creates a secret-shared backup of K_{master} such that the individual shares can be stored with several low-trust entities that are unlikely to collude [31].

B. Electronic Consent Request

With the present key management design, the healthcare facility must ask the patient (through their PPD) for a decryption key whenever it needs to process some monitoring data. This is reminiscent of the notorious GDPR mandate to get the data owner's consent. And the data key management can indeed be enhanced with a formal consent-requesting feature. When the processing application needs to decrypt one or more data records, such that it identifies the encryption key, resp. the intermediate secret value by a metadata string pair ($meta_{K_{data}}$, $meta$), it computes a request message Req that encodes ($meta_{K_{data}}$, $meta$), as well as a human-readable consent request, characterizing the data to be accessed, declaring the intent of the processing and how long the data will stay decrypted. Req is then digitally signed and transferred to the PPD, which verifies the signature, and prompts the patient for feedback, displaying the consent request message. The PPD may log the signed request as a non-repudiable evidence that such a request has been made to the patient. For this, the PPD must be securely bootstrapped with the digital signature key, which can be done in parallel with the platform credential bootstrapping.

V. Conclusions

In this paper, we present a complete key management solution for an IoMT patient monitoring system. The key management is modular, composed of a platform key management layer, which establishes ad-hoc, point-to-point secure channels between devices in the IoMT system, and of a data key management layer, which provisions keys for end-to-end encryption of patient data, with help of the platform key management's point-to-point security.

The platform key management enables point-to-point secure communications channels between devices of the IoMT monitoring platform and is a pre-requisite for the data key management. The function of the platform key management is proposed for data security in this paper, but its capabilities are much more diverse. In a similar fashion, it could support security functions such as local distributed computing at the edge, secure local intrusion detection, and more. Furthermore, the ability to develop a distributed platform key management domain is explored which would further improve the scalability as well as resiliency against attackers.

The data key management introduces a big shift in paradigm, giving the patient full control over the data protection keys to a patient, greatly reducing the degree to which a patient needs to trust the healthcare provider. It additionally empowers the patient by hardwiring the consent granting of GDPR into technology, the first design of its kind, to the best of our knowledge. At the same time, the design is remarkably scalable; the PPD only stores the master key and no other state, it can thus serve a practically unlimited number of sensors from an unlimited number of healthcare providers. On the other side, the tree-based key derivation facilitates data access to be extremely efficient at configurable scales.

Finally, the complete solution, as well as each of its parts, easily generalize to other IoT applications.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Direction," Future Generation Computation Systems, vol. 29, pp. 1645-1660, 2013.

[2] I. E. Essop et al., "Generating Datasets for Anomaly-Based' Intrusion Detection Systems in IoT and Industrial IoT Networks," Sensors, vol. 21, no. 4, 1528, 2021.

[3] D. Koutras et al., "Security in IoMT Communications: A Survey," Sensors, pp. 1-49, 2020.

[4] M. Neyja et al., "An IoT-based E-Health Monitoring System Using ECG Signal," IEEE Global Communications Conference (GLOBECOM), Singapora, Singapore, pp. 1-6, Dec. 2017.

[5] G. Hatzivasilis et al., "Review of Security and Privacy for the Internet of Medical Things (IoMT)," International Conference on Distributed Computing in Sensor Systems (DCOSS), Santorini, Greece, pp. 457-464, May 2019.

[6] M. Papaioannou et al., "A Survey on Security Threats and Countermeasures in Internet of Medical Things (IoMT)," Transactions on Emerging Telecommunications Technologies, vol. 4049, Wiley, pp. 1-15, 2020.

[7] M. Karageorgou, G. Mantas, I. Essop, J. Rodriguez, and D. Lymberopoulos, "Cybersecurity Attacks on Medical IoT Devices for Smart City Healthcare Services," IoT Technologies in Smart-Cities: From Sensors to Big Data, Security and Trust, F. Al-Turjman and M. Imran, Eds., IET, pp. 171-187, 2020.

[8] P. Voigt, and A. von dem Bussche, "The EU General Data Protection Regulation (GDPR): A Practical Guide," 1st edition, Springer, 2017.

[9] V. Karantounias, C. O'Donoghue, and O. Proust, "COVID-19 Questions & Answers - Coronavirus Emergency vs. GDPR Security Standards," MedTech, 10 April 2020. [Online]. Available: https://www.medtecheurope.org/wp-content/uploads/2020/03/20200410_MedTechEurope_COVID-19-vs.-GDPR-security-standardsQA_FINAL.pdf. [Accessed 25 May 2021].

[10] A. Mathur et al., "A Secure End-to-End IoT Solution," Sensors and Actuators A: Physical, vol. 263, no. 15, pp. 291-299, 2017.

[11] B. Mukherjee, R. L. Neupane, and P. Calyam, "End-to-End IoT Security Middleware for Cloud-Fog Communication," IEEE International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, USA, pp. 151-156, Jun. 2017.

[12] S. R. Moosavi et al., "End-to-End Security Scheme for Mobility Enabled Healthcare Internet of Things," Future Generation Computer Systems, vol. 64, pp. 108-124, 2016.

[13] Moore4Medical.eu, "Measurements at a Distance: The Missing Link to Chronic Monitoring," Moore4Medical, 2020. [Online]. Available: https://moore4medical.eu/continuous_monitoring. [Accessed 25 May 2021].

[14] P. P. Ray, D. Dash, and N. Kumar, "Sensors for Internet of Medical Things: State-of-the-Art, Security and Privacy Issues, Challenges and Future Directions," Computer Communications, vol. 160, pp. 111-131, 2020.

[15] M. A. Bouazzouni, E. Conchon, and F. Peyrard, "Trusted Mobile Computing: An Overview of Existing Solutions," Future Generation Computer Systems, vol. 28, pp. 596-612, 2018.

[16] C. Boyd, A. Mathuria, and D. Stebila, "Protocols for Authentication and Key Establishment," 2nd edition, Springer, 2020.

[17] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "The Kerberos Network Authentication Service (V5)," RFC 4120, Technical Report, 2005.

[18] T. Hardjono, "Kerberos for Internet-of-Things," IETF89, 2014.

[19] M. Sethi, B. Sarikaya, and D. Garcia-Carrillo, "Secure IoT Bootstrapping: A Survey," draft-irtf-t2trg-secure-bootstrapping-00, Technical Report, April 2021 (work in progress).

[20] J. Hong et al., "IoT Edge Challenges and Functions," draft-irtf-t2trg-iot-edge-02, Technical Report, May 2021 (work in progress).

[21] D. Boneh, and V. Shoup, "A Graduate Course in Applied Cryptography (v0.5)," 2020.

[22] L. Chen, D. Gollman, and C. Mitchell, "Key Distribution without Individual Trusted Authentication Servers," IEEE Computer Security Foundations Workshop (CSFW), Country Kerry, Ireland, pp. 30-36, Jun. 1995.

[23] L. Gong, "Increasing Availability and Security of an Authentication Service," IEEE Journal on Selected Areas in Communications, vol. 11, no. 5, pp. 657-662, 1993.

[24] A. Bogdanov, and A. Rosen, "Pseudorandom Functions: Three Decades Later," Tutorials on the Foundations of Cryptography, Y. Lindell, Ed., Springer, pp. 79-158, 2017.

[25] M. Naor, B. Pinkas, and O. Reingold, "Distributed Pseudo-random Functions and KDCs," Proceedings of EUROCRYPT, Prague, Czech Republic, pp. 327-346, May 1999.

[26] D. Boneh, K. Lewi, H. Montgomery, and A. Raghunathan, "Key Homomorphic PRFs and Their Applications," Proceedings of CRYPTO, Santa Barbara, CA, USA, pp. 410-428, Aug. 2013.

[27] A. Banerjee, and C. Peikert, "New and Improved Key-Homomorphic Pseudorandom Functions," Proceedings of CRYPTO, Santa Barbara, CA, USA, pp. 353-370, Aug. 2014.

[28] S. Kim, "Key-Homomorphic Pseudorandom Functions from LWE with a Small Modulus," Proceedings of EUROCRYPT, Zagreb, Croatia, pp. 576-607, Mar. 2020.

[29] K. Angrishi, "Turning Internet of Things (IoT) into Internet of Vulnerabilities (IoV): IoT Botnets," arXiv:1702.03681 [cs.NI], Feb. 2017.

[30] S. Herwig, K. Harvey, G. Hughey, R. Roberts, and D. Levin, "Measurement and Analysis of Hajime, A Peer-to-Peer IoT Botnet," Network and Distributed Systems Security (NDSS) Symposium, San Diego, CA, USA, pp. 1-15, Feb. 2019.

[31] A. Shamir, "How to Share a Secret," Communications of the ACM, vol. 22, no. 11, pp. 612-613, 1979.