# Reliability for Smart Healthcare: A Network Slicing Perspective

Andressa Vergütz, Guevara Noubir, and Michele Nogueira

## Abstract

Pursuing improvements in the healthcare system is mandatory for its efficiency and cost reduction. The fast popularization of implantable and wearable sensors promotes the diversity of healthcare applications and services, ranging from real-time and critical care monitoring to telemedicine. For smart healthcare (s-health), reliability plays an essential role, given the sensitivity of its data and services. In this article, we envision an architecture based on network slicing that can provide reliability for s-health applications and services. The architecture relies on fingerprinting healthcare applications to quickly customize resources and meet the level of reliability required for each s-health application. A fingerprinting study case is presented for s-health, based on a dataset containing real traffic. Results show that application fingerprinting reaches 90 percent accuracy, assisting in network customization. Finally, we discuss the main open issues and opportunities that network slicing technology provides for s-health.

## Introduction

Smart healthcare (s-health) is gaining significant attention from academia, industry, and the healthcare community [1]. S-health is essential to patients and clinics/hospitals, since it improves treatments, enhances patients' quality of life, and reduces costs, allowing the timely and ubiquitous provision of services and applications by information and communication technologies. Forecasts show a growing market for s-health of 24.1 percent (Smart healthcare market 2019 global industry analysis; http://www.theexpresswire.com/pressrelease/Smart-Healtcare-Market-Research-2019-Business-Opportunity-Global-Trend-Future-Growt-Key-Findings-and-Forecast-to-2022_10229596, accessed January 2020). All the advantages come with the cost of a significant increase in network traffic given the massive amount of sensed data and new applications. This causes high delays and a high level of losses, making it harder to achieve the high reliability required for s-health applications [2].

S-health applications and services are highly sensitive to network failures. For instance, in 2015, a network failure at Hillingdon Hospital, London, caused a severe loss of connectivity, which prevented the access of information necessary to treat patients (Network failure crashed frontline services at London hospital; https://www.computerweekly.com/news/4500247512/Network-failure-crashed-frontline-services-at-London-hospital, accessed January 2020). From the perspective of an ill patient with chronic disease, network failures can be disastrous for the patient and the hospital. Furthermore, regulatory institutions (e.g., the U.S. Food and Drug Administration) force the development of reliable health devices and applications [3] by acts such as the Health Insurance Portability and Accountability Act.

However, given the high level of reliability required by s-health, conventional network mechanisms (e.g., priority queues) become unsuitable due to traffic diversity and density [1, 4–6]. The variety of s-health applications (e.g., telemedicine, critical care monitoring, and physical activities) makes it even more complicated since each application has its requirements [5]. Furthermore, from the network traffic perspective, it is hard to differentiate these requirements and make efficient network decisions since s-health traffic is mixed with a vast amount of general network traffic (e.g., social networks, video streaming). Hence, it is indispensable to extract features and effectively analyze the network traffic to adapt network resources auonomously.

We advocate for the emerging concept of network slicing as a promising way to handle and provide reliability for s-health applications. Network slicing allows the virtualization of the physical network into virtual isolated subnetworks, offering flexibility, fast adaptation, and low costs. The benefits of network slicing for s-health comprise the autonomous analysis of network traffic and the adaptation of network resources to specific requirements, such as s-health reliability [5]. Hence, we introduce FLIPER, a Framework for Fingerprint s-Health Apps Traffic and Providing Network Resource Slicing. FLIPER automatically fingerprints s-health applications based on network traffic behavior and provides network resources autonomously by network slicing. Different from our previous work [7], FLIPER aims at achieving reliability for s-health applications, where each network slice receives specific resources according to the application requirements. FLIPER fingerprints applications to assist in the network slicing management.

Next, we detail the FLIPER framework, highlighting the benefits of network slicing for s-health. Then we present a background on network traffic analysis techniques to identify the traffic of s-health applications. Finally, we discuss the opportunities and open issues of network slicing for s-health and conclude the article.

*Andressa Vergütz and Michele Nogueira are with the Federal University of Paraná; Guevara Noubir is with Northeastern University, Boston.*

Digital Object Identifier:
10.1109/MNET.011.1900458



0890-8044/20/$25.00 © 2020 IEEE

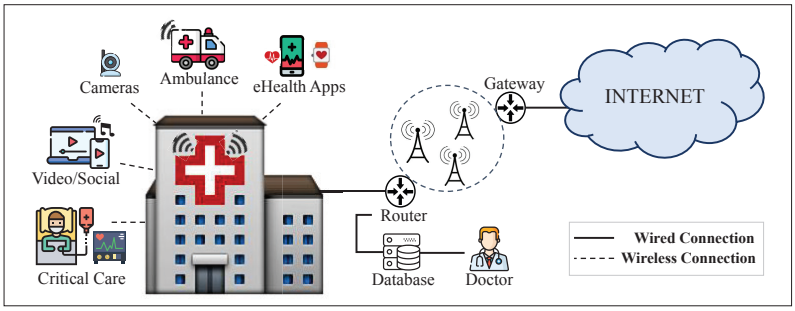


FIGURE 1. Smart hospital scenario.

## A Network-Slicing-Based Framework for Smart Healthcare

This section presents FLIPER. It complements existing works that use resources following a fault-tolerant perspective [4]. Different from those works, FLIPER employs the network slicing paradigm to provide automation, customization, and on-demand resource allocation based on fingerprinting analysis. Therefore, we contextualize the framework giving an overview of the addressed scenario, and we discuss the benefits of network slicing for s-health applications.

### Overview of Smart Health Scenario

The FLIPER framework works in the context of a smart hospital network infrastructure composed of different devices and supporting multiple types of applications, as illustrated in Fig. 1. The network infrastructure is heterogeneous in terms of hardware, software, and resource capabilities, and the devices range from wearables (e.g., smart glasses, smart watches, and cardiac sensors) to video surveillance cameras, desktops, servers, routers, and others. Wearable devices collect data and transmit it through a gateway to the Internet and cloud. Among the devices, there are wearable sensors continuously monitoring users' vital data (e.g., blood pressure). Healthcare practitioners have access to users' data through s-health applications, and they can employ data to clinical diagnosis or emergency medical response.

S-health encompasses different e-health (digital health) applications, consisting of any Internet applications focused on providing better conditions to the clinical processes and the treatment of patients. Hence, s-health involves a higher range of applications, including extreme critical care monitoring, telemedicine, remote surgery, and others [6]. Each application has specific requirements [8], such as extreme critical care applications that monitor patient health and need to react immediately, requiring ultra-reliable communication. Remote surgery can be mobile in some scenarios as in ambulances for disaster response. Hence, they need low latency and high reliability. Telemedicine for remote areas requires broadband connectivity. Natural disasters can involve many people with a massive number of wearable devices, requiring the overall management of density and scalability.

Hence, analyzing network data traffic based on information from the network layer (e.g., flow volume, packet size, inter-packet time) can assist in efficient reliability management. This analysis supports the creation of virtual slices according to the application requirements and the provision of network resources to better achieve reliability. Thus, in our conception, the framework can be positioned in a router serving as a gateway for a given network. Although such data from the network layer is not directly linked to the applications, it provides insights about the network traffic based on its behavior. For instance, wearable sensors that collect vital signs send periodic data, unlike other applications, such as social media applications, that generate traffic at a constant rate. Therefore, it is possible to differentiate the application traffic, which assists in network slicing management.

### Detailing FLIPER Architecture

FLIPER comprises four main modules: *Pre-processing*, *Feature Extraction*, *Fingerprinting*, and *Network Slicing Configuration*. Figure 2 shows the four modules with its components, where the input consists of the network data collected to extract the application information and to create network slices. The first module filters the network traffic according to certain features of reference (a.k.a. ground truth information). Its main goal lies in associating the traffic with the devices as a first step. Based on the ground truth information, FLIPER filters the network traffic per device, assisting the functionalities of the other modules. The second module selects and extracts the information from the network layer (the values for the employed features) and the statistical properties from the traffic of each device. The third module gets the labeled data and creates data for training, validation, and tests, and then it applies machine learning (ML) algorithms to measure its performance by metrics as accuracy and precision. Finally, after fingerprinting applications, the last module manages network slices following specific configurations for each s-health application. In Fig. 2, FLIPER follows the requirements of three different s-health applications since it aims to achieve s-health requirements.

**Pre-Processing Module:** It handles the network traffic in two components: the collection of ground truth information and data processing for each device. Together, they create subsets of network traffic belonging to each device ($D$). FLIPER uses the medium access control (MAC) address as ground truth to correctly indicate to which device each traffic flow belongs. Thus, based on the ground truth information, FLIPER splits the network traffic and labels according to each device. Hence, FLIPER creates a subset of network data for each class of devices, denoted by a set $D = \{d_1, d_2, ..., d_n\}$, where each $d$ means a specific device class with its network traffic.

**Feature Extraction Module:** It receives as input the set of device classes $D$. Afterward, for each class $d_i \in D$, FLIPER selects and extracts the features from the network traffic, defined as a set $F$ of $m$ network features, $F = \{f_1, f_2, ..., f_m\}$ that belongs to each class. FLIPER employs packet size and flow volume as side-channel features once these two features play a relevant role in the traffic of s-health applications. For instance, s-health applications related to monitoring physiological data yield small packet size (e.g., around

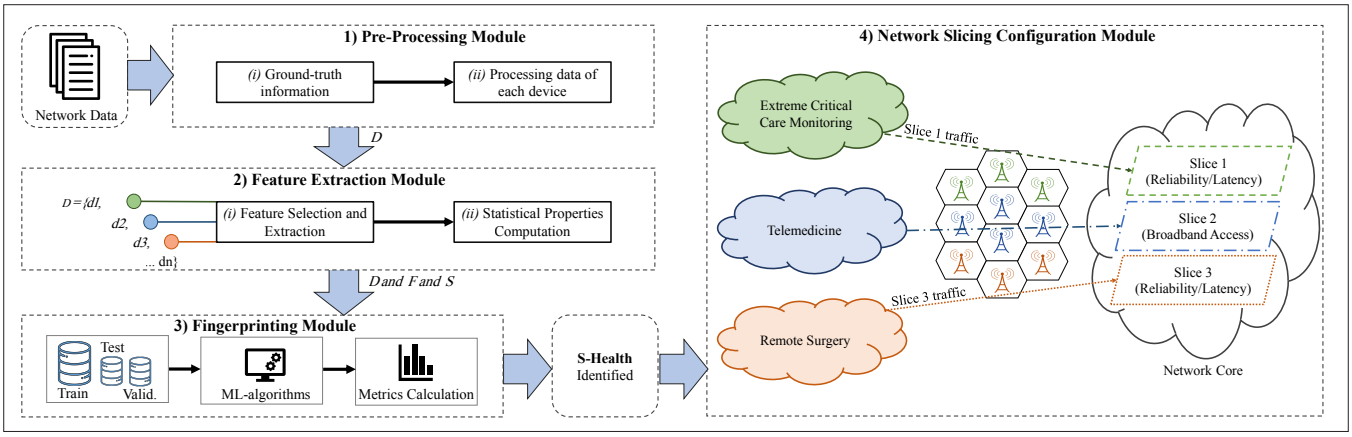


FIGURE 2. FLIPER architecture.

50 bytes), and generate a small volume compared to other applications. To differentiate the classes of devices, after extracting the network features, FLIPER computes their statistical properties. These properties consist of measures such as average, variance, and others. For each set of features $F$, FLIPER calculates a set of $x$ statistical measures, $S = \{s_1, s_2, ..., s_x\}$, where $D \ni F \ni S$. For instance, for the packet size feature, FLIPER computes the maximum (s1), minimum (s2), average (s3), and variance (s4). FLIPER follows this procedure for all employed statistical measures. Then this module offers to the third one the set of network features $F$ and the set of statistical properties $S$ for each class of device $D$.

**Fingerprinting Module:** This uses distinct types of ML algorithms to fingerprint different s-health applications, for example, the Random Forest based on decision tree rules. It receives as input the network traffic features $F$ and statistical properties $S$ of each class of device. As each class is labeled, FLIPER merges them in a final file, called training data. It supports unbalanced data (i.e., a dataset containing applications with different amounts of data). The fingerprinting of applications follows the holdout method, where the training data is divided into testing and validation data. After creating the final file, FLIPER applies the ML algorithms and computes performance metrics (e.g., accuracy, recall) to fingerprint the classes of applications. The final output lies in a table with the percentage values for the metrics to each identified class, including the classes of s-health applications.

**Network Slicing Configuration Module:** This receives as input the results of the application fingerprinting and allocates network resources to achieve the requirements of each application. For creating slices, FLIPER verifies a table that contains the specific requirements of the applications, including s-health. According to the requirements, it requests of the Virtual Infrastructure Manager (VIM) the creation of a network slice (NS), one NS for each fingerprinted application. VIM is a common component of network virtualization, controlling and managing the network function virtualization (NFV). VIM verifies the network resources available and requests of the Network Slice Management Function (NSMF) the slice creation. NSMF manages and selects the appropriate network slice based on the application requirements. The application receives the network resources first. When there is not enough available network resources, VIM analyzes the type of running slice to drop or share its resources. VIM and NSMF verify the availability of resources and decide about the creation of NSs.

FLIPER creates the slices following the application requirements defined by Next Generation Mobile Network Alliance (NGMN) and 3rd Generation Partnership Project (3GPP) standardization efforts for 5G networks. These efforts consider scenarios such as natural disasters and extreme critical care monitoring [8, 9]. Hence, in Fig. 2, each application receives a type of NS according to its requirements. For instance, the extreme critical care monitoring receives a slice configuration with high reliability and low latency (ultra-reliable low-latency communications, URLLC, type). Telemedicine for patient monitoring receives a slice with broadband connectivity (enhanced mobile broadband, eMBB, slice), while remote surgery in an ambulance for disaster response situations follows its restrict requirements in terms of high reliability, low latency, and security (URLLC type). Moreover, pre-establishing slices is crucial to achieve end-to-end low latency. The next subsection explains how network slicing benefits s-health and the types of slice.

## Benefits of Network Slicing for S-Health

Network slicing aims at improving the quality of service for applications. S-health requires efficient networking capabilities to provide low latency, low loss rate, and high reliability. Thus, network slicing offers service customization and isolation on physical network infrastructure, enabling the logical as well as physical separation of network resources [5]. The main idea lies in a single physical piece of network infrastructure able to cost-effectively deliver multiple logical networks (slices) over the same network infrastructure. The 3GPP 5G standardization establishes three main slice/service types (SSTs): eMBB, URLLC, and massive Internet of Things (mIoT) [9]. These types of slices encompass several contexts, including s-health, dense urban, and others [8]. In the s-health context, extreme critical care applications belong to URLLC since they requires packet loss as low as 1 out of every 10,000 packets and 1 ms latency [5].

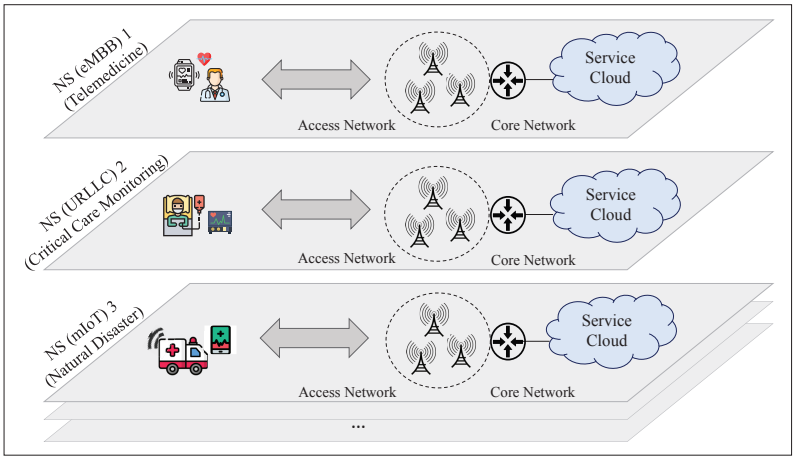


FIGURE 3. Example of network slices for s-health.

Network slicing employs software defined networking (SDN) and NFV to slice a network into logically isolated network slices [5]. Each slice may have certain properties, such as latency and high reliability. High reliability means successfully transmitting data packets without extrapolating the maximum latency supported by the application. Communication and buffer resources along a path are allocated according to the expected data arrival from the s-health devices to guarantee low latency. Figure 3 shows a network slicing perspective for s-health. Slice 1 (NS-1) involves telemedicine applications monitoring patients in remote areas. Thus, its slice type is eMBB with broadband connectivity. NS-2 contains network resources to achieve low latency and high reliability since it is established for extreme critical care monitoring applications (URLLC). NS-3 is established for emergency response in natural disasters. In this case, massive devices from the survivors send messages to the hospital. This scenario requires a slice type of mIoT to manage the number of devices. These three NSs represent a smart hospital with different use cases and types of traffic, each with its specific requirements. Due to the diversity and density of applications and traffic, fingerprinting assists the network slicing management.

Another network slicing advantage is the flexibility to provide virtual analytic data and automation. Virtual analytic data consists of analyzing the network traffic and its behavior, and automation provides the network services and resources necessary according to the network behavior and performance. Hence, data analytics complement automation. To observe network traffic, techniques of data analytics can be applied, such as fingerprinting. However, once network slicing is a new paradigm, there is no standard indicated to do such traffic analysis. In this article, we advocate fingerprinting techniques based on ML algorithms since they quickly learn traffic features and behaviors. Fingerprinting s-health traffic can assist the automation of decisions about network resources adaptation. With this purpose, the next section presents a study case about fingerprinting s-health traffic and the results from the implementation of this case study.

## A Case Study of S-Health Fingerprinting

The ultimate goal of our network-slicing-based framework is to fulfill the reliability requirements of s-health applications through data analytics on network traffic features and behaviors. For data analytics, we present an s-health case study following the background of the main fingerprinting techniques.

### Background on Fingerprinting s-Health

Fingerprinting the legitimate traffic of an active s-health application allows us to provide reliability services, such as resource allocation and fast fault recovery. However, this is a difficult task. First, researchers must deal with an increasing amount of traffic as well as equally increasing transmission rates [10]. Researchers are looking for lightweight algorithms with as low computational requirements as possible to cope with such high speed and volume. Moreover, network application developers further hamper the task of identifying applications by any available technique that hides traffic and confuses network operators, like data encryption and encapsulation. Therefore, there is a need for novel and unexpected ways of identifying traffic applications, even more for s-health applications given their sensitive data.

We have investigated the current methods from the literature to identify the traffic of an application. Typically, researchers follow four different approaches to analyze network traffic and classify an application: *port-based approach*, *payload-based approach*, *behavioral-based approach*, and *statistical-based approach*. Although port-based approach classification is a fast and straightforward method, several studies have shown that it performs poorly. As the protocols are assigned to well-known transport layers ports by IANA, this approach extracts the transport layer port from the packet header and looks it up in the table containing the port-application associations. However, this can be inaccurate and unreliable, because current applications may hide traffic behind ports of other protocols or behind well-known port numbers (e.g., TCP port 80), or use dynamic ports [11].

In contrast, the payload-based approach is a usual approach to identify applications by payload packet analysis [12]. It matches a deterministic set of signatures or regular expressions against packet payload [10]. Packet payload is searched for known patterns, keywords, or regular expressions, which are characteristics of a given protocol or application. This approach has reached high accuracy in identification. However, analyzing the packet payload on such health information violates users' privacy [11].

The behavioral-based approach observes the total traffic received by a host or by an endpoint in networks [10]. Thus, all host traffic is captured, and the host application signatures are compared to the captured profile, and then the traffic flow is classified. Generated pattern traffic is observed (e.g., how many hosts are connected, with which transport layer protocol) to identify the application running on the target host. However, this approach needs previous knowledge, such as signatures from the traffic.

Considerable attention has been given in data mining and ML algorithms. ML algorithms are based on a statistical-based approach. Instead of analyzing payload, this approach considers flow-level measurements, for example, packet size, inter-packet time, flow volume, and statistics measurements (e.g., mean and max packet size). However, most studies in the literature use this approach to identify common Internet and mobile applications [13] and identify IoT devices [14]. We have found only one study that addresses s-health applications [11]. Since s-health has a lot of requirements, such as critical care monitoring that carries information about the vital signs of patients, they need special attention [2]. Thus, fingerprinting s-health applications enables the network management to achieve these requirements.

## Implementing S-Health Fingerprinting

FLIPER follows four main modules leading its implementation. Our framework works in the context of a smart hospital environment with several heterogeneous devices sharing data. The representative *IoT Traffic Analysis* dataset [14] serves as input for fingerprinting in performance evaluation. For the results, we employ a dataset containing network traffic from 20 days (September 22, 2016 to October 12, 2016). Its contains records of real network traffic from different types of devices, including monitoring cameras, baby monitors, wearable sensors (e.g., Sleep Sensor and Blipcare blood pressure), laptops, smartphones, hubs/controllers, environment light sensors (e.g., LIFX light bulb), movement sensors (e.g., Belkin Motion), and portable speaker assistance (e.g., Triby Speaker and Amazon Echo). Among them, there are two healthcare monitors: Sleep Sensor and BlipcareBP. Our goal lies in fingerprinting the traffic from these two devices among the others.

The dataset is the basis for network traffic features and statistical properties extraction. The framework employs features such as packet size, flow volume, and inter-packet-time, extracted by Tshark and Cisco-Joy tools. The input file to fingerprint s-health applications is composed of these features and their associated capture time. The capture time encompasses the trace capture day from the dataset. We normalize the capture time to an interval from 1 to 20 days, and we extract packet sizes and inter-packet time from the header of the network layer. A flow means the packets of the data plane between sender and receiver that share key IP header information. A flow encompasses the 5-tuple information: same source and destination addresses, same source and destination ports, and the same protocol. Flow volume is the sum of all bytes sent from a source to a destination. We ignore packets destined to DNS servers and packets or flow volume with size equal to 0. Moreover, the MAC address of each device serves as the label of the training data (Pre-Processing Module).

For packet size and inter-packet time features, we compute the statistical properties *min*, *max*, *average*, and *variance* through the Numpy library from Python v2.7 (Feature Extraction Module). Such statistical properties are added to a file according to each network traffic feature. After statistical calculation, FLIPER creates the training and testing data using the input files. It applies the Random Forest classifier [13], since it is a multiclass classifier, making it suitable for tasks such as application and device fingerprinting. We use metrics such as accuracy, precision, and recall (Fingerprinting Module). These metrics assist in evaluating reliability. Their results support the network slicing, offering insights about network traffic and application requirements. High accuracy, precision, and recall result in efficient and correct creation of slices.

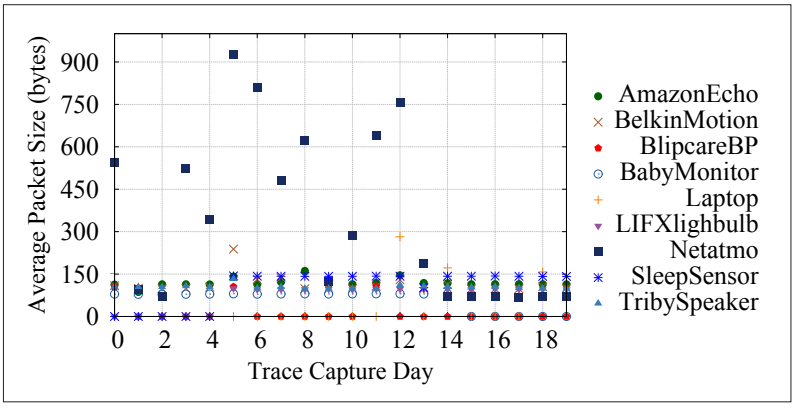


FIGURE 4. Average packet size vs. capture day.

Figure 4 presents the average packet size per device identified in the dataset. Wearable sensors (BlipcareBP and Sleep Sensor) have reached small values for average packet size. BlipcareBP presents a tiny amount of traffic since it shows an average packet size smaller than 100 bytes and only generated traffic for three days. Sleep Sensor reaches average packet size around 130 bytes. In contrast, Netatmo reaches more than 800 bytes in average packet size and the laptop 300 bytes, while other devices present similar behavior. This behavior indicates that the dataset contains unbalanced data (i.e., dataset presents a diverse amount of data), which is a typical case for s-health. It is harder to fingerprint the smaller traffic.

Figures 5 and 6 show the Random Forest performance on application fingerprinting. The results encompass an overall macro average of nine classes, where each class corresponds to one device (i.e., nine types of devices). We run the classifier 50 and 100 times for the results in Figs. 5 and 6, respectively. In Fig. 5, Random Forest presents a stable macro average for accuracy, maintaining around 90 percent. Precision presents values from 73 to 90 percent, indicating a small amount of false positives. Recall presents values around 70 percent, with a lower number of false negatives. In the s-health context with ill patients, higher recall is more relevant than precision, since for recall it is essential to identify all ill patients, even if classifying a healthy patient as sick (FP situation). Considering the s-health devices from the dataset, FLIPER correctly fingerprints almost all traffic from the Sleep Sensor. Figure 6 shows the results for BlipcareBP once it has the smaller traffic from the dataset. Accuracy reaches values close to 100 percent, while precision and recall present values from 69 to 98 percent due to the small amount of traffic of BlipcareBP.

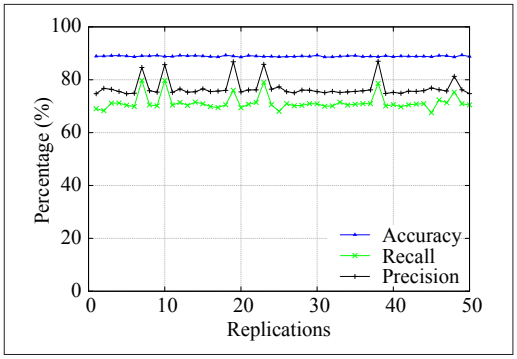


FIGURE 5. Fingerprinting classifier performance for all devices.

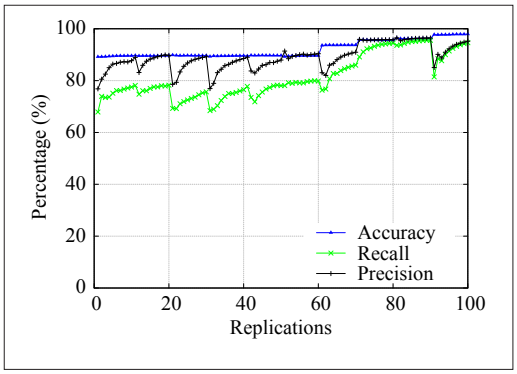


FIGURE 6. Fingerprinting classifier performance for BlipcareBP device.

## Open Issues and Opportunities

This section presents the main open issues and opportunities in the application of network slicing for s-health applications, and we highlight interesting future directions.

### Reliability

High reliability encompasses network communication without failures, data and services available as long as possible, and support for service operation. For instance, unreliable network quality can cause a high rate of packet loss and errors, which can lead to an erroneous diagnosis by a health professional [2]. However, ensuring reliability is complex, and it is not straightforward.

Hence, adopting network slicing for s-health would be beneficial. First, it enables the adaptation of network resources and services when a network failure occurs, since network slicing has automation and isolation characteristics. Data analytics of network traffic provided by network slicing jointly with fingerprinting techniques extract network behaviors to assist in resource allocation. However, there is an issue related to failure detection because network slicing is not prepared to detect a network or data failure. Some service positioned in each slice detecting data failures could be a solution. Then, when detected, a failure data packet can be sent to another concurrent slice. Another issue lies in providing reliability by the slice infrastructure: should it be implemented as embedded slice mechanisms, like multipath transmission and queuing priority? Probably both mechanisms will have to cooperate, such as in the use of a URLLC slice configuration with queuing priority.

### Performance

S-health applications require performance. End-to-end delay is significant for health-related applications, supporting only 125 ms of latency. In the network slicing context, critical care applications only support 1 ms, and it is very difficult to reach such latency in traditional networks. Virtual slices with network resources allocation on demand provide specific services to reach such latency. However, the slice deployment time and slice transition time (e.g., the time to change the applications for another slice) can be critical for some s-health applications. Such cases can be supported by special mechanisms. These mechanisms can include the creation of functions shared by multiple slices and/or deploying a priori specific slices and keeping them in a "frozen" state, reducing the use of resources.

Fingerprinting the network traffic and knowing necessary services is essential to assist end-to-end latency. However, two critical issues are the small amount of traffic and ground truth information definition. New approaches are fundamental to fingerprint an s-health application with higher precision, even with a small amount of traffic. Specif ic ML techniques for unbalanced data can be a solution. The ground truth is used to correctly fingerprint the application. Studies have applied the MAC address or payload inspection techniques to collect ground truth. Nonetheless, sometimes the MAC addresses of devices are unavailable, and the payload inspection techniques violate user privacy. Hence, advances are indispensable.

### Security

Security and privacy are undoubtedly crucial for s-health because it uses location, personal, and context information of users. S-health applications are susceptible to various types of attacks (e.g., patient tracking, side-channel attacks, and denial of service). Attackers can use the data obtained through network data analytics and fingerprinting techniques (ML algorithms). Side-channel attacks can benefit from network traffic to infer information about the users. ML algorithms can solve challenges in the security of network slicing, such as to control the network slice behavior. When the behavior changes, the system triggers an alert. However, attackers can mimic the s-health traffic behavior. In this case, authentication techniques and fingerprinting would jointly improve security.

Another issue lies in network slicing isolation. An opportunity refers to inter-slice isolation. It would assist in preventing an attack on a slice to affect other network slices and the attack propagation when using shared functions by the "cascade effect." Moreover, network slices might have different security levels and policies since different providers could manage them. The challenge is how to enforce network slice security when a network function or slice is compromised and infrastructure from different providers is in use. Similar security mechanisms to different infrastructure providers are probably required for any provider to take preventive action.

## Cost

S-health applications have cost and complexity restrictions. There are concerns related to the trade-off between complexity and application latency. Although network slicing creates slices on demand that are customized according to the needs of specific applications, the slice creation, management, and isolation can highly increase complexity and cost. The number of slices operating in network infrastructure is expected to be huge, which makes network management difficult. The management issues grow not only with the number of slices, but also with their complexity. Fog devices and pre-created slices can assist in the management and configuration of slices to decrease latency and cost.

## Conclusion

In this article, we present a network-slicing-based framework for smart healthcare (s-health). Such framework can indeed boost s-health applications reliability by efficiently handling the amount of data generated by wearable sensors as well as other devices, and create specific network slices to achieve the application requirements. Network slicing for s-health is attractive because it assists in network resource allocation according to the specific requirements (e.g., high reliability and low latency). The framework fingerprints network traffic with 90 percent accuracy. In the future, we will test other fingerprinting approaches to improve the accuracy and implement the network slices.

## Acknowledgments

The authors thank the agencies CAPES, CNPq, and the joint NSF and RNP HealthSense project, grant #99/2017.

## References

[1] D. He *et al.*, "Privacy in the Internet of Things for Smart Healthcare," *IEEE Commun. Mag.*, vol. 56, no. 4, Apr. 2018, pp. 38–44.
[2] I. De la Torre Díez *et al.*, "Systematic Review About QoS and QoE in Telemedicine and eHealth Services and Applications," *J. Medical Sys.*, vol. 42, no. 10, 2018, p. 182.
[3] FDA, "Digital Health Innovation Action Plan," Tech. Rep., Feb. 2018, accessed July, 2019.
[4] K. Wang *et al.*, "Mobile Big Data Fault-Tolerant Processing for eHealth Networks," *IEEE Network*, vol. 30, no. 1, Jan./Feb. 2016, pp. 36–42.
[5] I. Afolabi *et al.*, "Network Slicing & Softwarization: A Survey on Principles, Enabling Technologies & Solutions," *IEEE Commun. Surveys & Tutorials*, 2018.
[6] A. A. Abdellatif *et al.*, "Edge Computing for Smart Health: Context-Aware Approaches, Opportunities, and Challenges," *IEEE Network*, vol. 33, no. 3, May/June 2019, pp. 196–203.
[7] A. Vergütz *et al.*, "A Method for Identifying Ehealth Applications Using Sidechannel Information," *Proc. IEEE GLOBECOM*, 2019, pp. 1–6.
[8] NGMN Alliance, "Description of Network Slicing Concept, NGMN 5G P1 Requirements & Architecture, Work Stream End-to-End Architecture," NGMN 5G P, vol. 1, Jan. 2016.
[9] 3GPP, "System Architecture for the 5G System," 3GPP TS 23.501 v. 15.2.0, Release 15," 3GPP 5G Initiative, Tech. Rep., June 2018, accessed July, 2019.
[10] H. Kim *et al.*, "Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices," *Proc. ACM CoNEXT*, 2008, p. 11.
[11] M. Grajzer *et al.*, "A Multi-Classification Approach for the Detection and Identification of eHealth Applications," *Computer Commun. Net.*, IEEE, 2012, pp. 1–6.
[12] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and Future Directions in Traffic Classification," *IEEE Network*, vol. 26, no. 1, 2012.
[13] V. F. Taylor et al., "Robust Smartphone App Identification via Encrypted Network Traffic Analysis," *IEEE Trans. Info. Forensics Security*, vol. 13, no. 1, 2018, pp. 63–78.
[14] A. Sivanathan *et al.*, "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," *IEEE Trans. Mobile Comp.*, 2018.

## Biographies

Andressa Vergütz [S] is a Ph.D. candidate in computer science at Federal University of Paraná, Brazil. Her research interests include problems related to wireless body networks, IoT, security, and data analysis. She is a member of the Wireless and Advanced Networks research team. She is a student member of the Brazilian Computer Society.

Guevara Noubir is a professor in the Khoury College of Computer Sciences at Northeastern University and the director of the Cybersecurity Graduate Program. His research spans a range of problems in the theory and practice of privacy, security, and robustness in networked systems. He is particularly interested in mobile and wireless systems security. Over the years, he has built interest and expertise in wireless interfacing with biological systems, and the leveraging of synthetic biology.

Michele Nogueira [SM] is an associate professor of computer science at Federal University of Paraná. She holds as doctorate in computer science from the Sorbonne University — Pierre et Marie Curie, Laboratoire d'Informatique de Paris VI. Her research interests include wireless networks, security, and dependability. She is a Senior Member of ACM.