NausicaApp: a Hybrid Decentralized Approach to Managing Covid-19 Pandemic at Campus Premises

Giovanni Marotta, Fabrizio Billeci, Giuseppe Criscione, Fabio Merola, Giuseppe Pappalardo, Emiliano Tramontana
Department of Maths and Computer Science
University of Catania
Catania, Italy
giovanni.marotta@phd.unict.it, fabrizio.billeci@studium.unict.it, giuseppe.criscione@studium.unict.it
W82000188@studium.unict.it, pappalardo@dmi.unict.it, tramontana@dmi.unict.it

2020 Asia Conference on Computers and Communications (ACCC) | 978-1-7281-8967-3/20/$31.00 ©2020 IEEE | DOI: 10.1109/ACCC51160.2020.9347901

***Abstract*—Several contact-tracing app-based solutions have been proposed to alert app users who have previously encountered a Covid-19-positive user. In most countries, a state-sponsored app has been made available to let citizens who install it on their phone monitor inter-personal contacts.**

The particular scenario considered in this work aims to support tracing within a university campus. For this, an alternative solution is proposed, based on a hybrid decentralized approach, that traces people who have been present in the same environment within some meaningful time interval. Thus, tracing goes beyond direct contact between persons. Our solution can also monitor crowd gathering on campus premises.

The proposed Android app senses surrounding WiFi signals and uses them to obtain the user's absolute location. The app then securely sends an anonymized presence data object to the server. Thanks to data thus gathered, as soon as a user has reported herself as Covid-19-positive, all apps will be alerted by the server and receive anonymous data to determine whether their users happened to be in the same environment as the Covid-positive one.

We believe our approach to be both effective, for it eschews weaknesses and limitations of Bluetooth-based solutions, and viable, for the experiments reported have proved WiFi sensing-based localization to be accurate enough.

***Index Terms*—contact tracing, localization, data analysis, mobile applications, decentralization, distributed system, privacy**

I. Introduction

Since the outbreak of the Covid-19 pandemic, one of the main issues to be faced in working environments has been to ensure tracing of contacts between potentially Covid-19 positive individuals, while at the same time preserving their privacy. Contact tracing is crucial to providing the necessary alerts, as the pandemic spreads, so that a healthy environment can be preserved.

Many countries have developed and distributed contact tracing solutions that differ along a variety of dimensions, including especially the balance struck between responding to the emergency with an effective pandemic surveillance, and upholding individual rights and liberties. Design choices may vary according to cultural and political background, especially to what can be referred to as third party trustworthiness (e.g. Ministry of Health).

Contact tracing solutions basically consist of two components: (a) mobile apps to be (voluntarily) installed on user personal devices, and (b) an administration server. The computational functionalities, communication channels and storage capabilities (required) of parts (a) and (b) depend on the adopted solution.

The spectrum of current solutions spreads from almost totally centralized ones, such as South Korean Corona100m[1], where a trusted server is in charge of storing and processing the data coming from user devices, to mostly decentralized ones, such as Italy's Immuni[2], where personal devices directly interact via Bluetooth when they are within range. Immuni apps exchange and store anonymized IDs and other location data, report infected cases and trace suspicious contacts, with the server just playing the role of an information relay among end-users. Centralized solutions, often based on the absolute location of their users, are mainly targeted at providing very efficient measures to follow up contact tracing of Covid-19 positive people (e.g., mobility patterns, contagion risks, location monitoring). They rely on the trustworthiness of central servers and authorities but commonly elicit privacy concerns [1]. In contrast, decentralized solutions, which most of the times depend on the users' relative location, cannot provide a global view of contact tracing but guarantee a deeper safeguard of individual privacy [2].

Although, in each country, only one app can access the underlying OS's support for contact tracing [3], industrial and academic research have come up with a plethora of proposals, with an eye to different deployment environments, in which underlying technologies, as well as use cases and specific needs may drive different architectural choices.

NausicaApp, the solution proposed in this paper, aims at supporting post-Covid-19 management of critical working conditions in campus environments, thus guaranteeing the continuity of university-level research and teaching in a context of shared social security, in compliance with current hygiene and health requirements. It is part of the wider NAUSICAA (New Approach for a UniverSIty Covid-resilient and Active Again) project at the University of Catania, aimed at addressing Covid-19 related concerns with a multidisciplinary

[1]http://www.korea.net/NewsFocus/Society/view?articleId=183129
[2]https://www.immuni.italia.it

© IEEE 2021. This article is free to access and download, along with rights for full text and data mining, re-use and analysis.

approach, spanning the fields of medicine, law and public space management.

The app, complete with support and back-end services, has been developed according to a hybrid decentralized design, to respond to multiple requirements: (1) the management of people flows and space occupation rates on campus; (2) the booking, with fair rotation, of lectures in attendance; (3) the tracing of direct and indirect contacts for epidemiological surveillance; (4) the aggregation of collected anonymized data, aimed at driving informed campus management decisions, and to be shared with local and national health authorities,

From an architectural point of view, NausicaApp is the client component of a system solution that is loosely based on the Decentralized Privacy-Preserving Proximity Tracing (DP3T) proposal [4], with which it shares the principle that storing and processing of direct contacts are the responsibility of the mobile apps. At the same time, however, our solution adopts a centralized communication infrastructure to host the data exchange between mobile devices and the administration server.

The standard campus WiFi enterprise infrastructure or, in adverse indoor conditions, an external service, is exploited for absolute localization, while data exchange can be indifferently carried out over the same WiFi infrastructure or cellular networks.

Experimental results, reported below, fully vindicate the adoption of WiFi sensing as an appropriate technique for device absolute localization in the target environment and scenario.

Typical security issues of DP3T-like systems [5] are mitigated by the design choice of avoiding use of the Bluetooth channel, which can make smartphones vulnerable to well-known attacks [6].

II. Related Works

DP3T

As previously mentioned, the DP3T proposal [4] leaves the ownership of the collected data with the end-user apps, while letting the central server only play the role of a relay system whenever users self-declare as Covid-19 positive. DP3T employs a dedicated Bluetooth app-to-app communication channel for direct exchange of proximity data, which is retained by each app in its local storage. Even though contact tracing is performed locally to the mobile apps, this decentralized solution exhibits some very interesting features, but also drawbacks mainly arising from resorting to Bluetooth.

Although the solution put forward here shares DP3T's decentralized approach to tracing direct and indirect contacts, it does not employ a direct Bluetooth app-to-app communication channel, and, as a result, avoids the security issues raised by Vaudenay [5].

In short, our main differences from the DP3T solution can be listed as follows, and will be more deeply discussed in Section III: (i) instead of Bluetooth, WiFi sensing is used to localize the currently active devices and collect presence data of individuals within campus locations and sites (e.g., lecture rooms, offices, libraries, etc.); (ii) ephemeral user IDs are not generated by the apps, but rather linked to session IDs managed by the back-end platform; (iii) suspicious contacts are not detected based on ephemeral individual IDs matching, but by means of a business logic which correlates coincidental places and times of presence data; (iv) presence histories of Covid-19 positive users can be used for both direct and indirect contact tracing, i.e., an alert can also be raised to warn people who have attended the same places (e.g. lecture halls), but at different times (within a meaningful interval).

EPIC

This is a solution designed and prototyped by Altuwaiyan et al. [7] for indoor environments. It uses hybrid short-range wireless technologies, namely WiFi and Bluetooth, with the aim to provide fine-grained response to direct-only contact tracing goals. EPIC is mostly centralized in that, even though localization data are collected and stored at client side, they are processed at server side to assess if contacts with infected users have occurred, and if these have to be regarded as critical. Because of this architectural choice, much emphasis is placed on cryptographics techniques.

In contrast, since our NausicaApp solution is decentralized with regard to contact tracing assessment and relies on strong deanonymization of user information, it can afford to make use of standard security protocols for data transmission over protected channels (e.g., WiFi enterprise infrastructure). Some potential residual vulnerabilities, still affecting NausicaApp despite deanonymization, are addressed by means of specific control logic built in the app design.

WifiTrace

As expressly stated in [8], "WifiTrace is a network-centric solution for contact tracing that relies on passive WiFi sensing with no client-side involvement". Like NausicaApp, it has been designed and prototyped for campus environments, and is mainly scoped for post-processing device trajectories and reconstructing on-campus people flows. It strongly relies on passively collected WiFi enterprise network logs. This approach assumes central and third-party authorities to be fully trusted.

Our solution does make use of some WiFi infrastructure configuration information to better characterize wireless fingerprints during the sensing and localization phase (essentially, in order to map APs to campus sites). However, it has been decided that network logs shall not be used to track users flow for privacy preserving reasons. Indeed, the same goal can be pursued by properly processing the fully deanonymized user data that are collected at the server side.

TraceTogether

Singapore's TraceTogether[3] relies on centralized server capabilities for the purposes of contact tracing and contagion risk advertisement to individuals who have come close to reported infected people. Even though relative localization data are

[3]https://www.tracetogether.gov.sg

collected at user side and cryptographically exchanged with the server, once a user is reported positive, both her and her contacts will be deanonymized by the server, owned by the Ministry of Health. This highly beneficial choice for the contact tracers does not guarantee the right to anonymity at all, and yet is considered absolutely legal in Singapore [9]. The adoption of a similar solution in Australia (i.e., COVIDsafe[4]) had to undertake some steps to address a few legitimate privacy concerns[5].

Although NausicaApp aims at providing support for epidemiological surveillance and tracking of infected people along with their contacts, such features can be obtained centrally only in terms of anonymized absolute localization data, which can also provide indirect contact tracing information that relative localization data cannot.

Co100

The South Korean solution Corona-100m, or Co100 for short, is by far the most centralized and the least privacy preserving among the ones analysed so far. GPS or cellular networks absolute localization data are collected by user devices and sent to the central authority with the aim of providing a publicly available website[6] which helps people to find out where (anonymized) infected people have been in the previous days. Also, deanonymized data belonging to confirmed Covid-19 patients can be immediately provided to health investigators[7].

It goes without saying that Co100 has very little in common with our solution in terms of technological and privacy choices. Only tracing of indirect contacts and follow-up support to health authorities can be loosely considered as common concepts, in that in NausicaApp everything is designed in order to provide user data anonymity and data confidentiality, yet guaranteeing data authenticity as discussed in Section IV.

III. Solution Architecture

A. Overview

NausicaApp is designed to ensure advanced functions in many areas as outlined earlier in the paper in Section I. The solutions proposed feature some original choices in the way localization information is built and potential proximity contacts are detected. The system architecture is based on a mixed client-to-client and client-server solution with ad-hoc client software in charge of data collection, storage and computing and the server only acting as a centralized relay and alert system. Indeed, no data matching processing is performed outside the personal devices, with a central server only responsible for relaying centrally verified information, every time an infected user has authorized the distribution of anonymous presence information to the rest of involved devices. App/server communication simply assumes user devices have access to the Internet (possibly, but not necessarily, leveraging the pre-existent campus-managed enterprise WiFi infrastructure), in fact making no use of any app-to-app direct channel (e.g., Bluetooth).

The central server can use the collected anonymized presence data to detect crowd gatherings in various spots, emit alarms and send alerts, as well as monitor people flows and occupation rates on the premises, so as to facilitate the smart management of potentially critical situations in a daily prevention routine.

An authorized campus representative may decide to share, under strict privacy-preserving policies, aggregated data with epidemiologists and health authorities for surveillance, contagion analyses and medical follow-ups.

B. Components

The proposed distributed system (see Figure 1) is based on a Firebase platform[8] featuring the Firebase Cloud Messaging (FCM)[9] cross-platform service and consists of four main components for building and sending notifications to FCM-enabled apps and receiving messages from them[10]:

1) a trusted server environment that supports the Firebase Admin SDK, and has the task to configure user authentication and addressing logic, as well as build the notifications that will be used by the FCM back-end. In our prototype system, the Spring Boot[11] framework, deployed on the Heroku Cloud Application Platform[12] is the chosen environment hosting the Firebase Admin SDK;
2) the FCM messaging service, which adds transmission functionalities to the Firebase Admin SDK: it is used to distribute notifications and messages to devices subscribed to the server platform and, conversely, to relay messages from the apps to the administration server;
3) the Android Transport Layer, which supports FCM;
4) FCM-enabled Android apps that receive notifications from the FCM messaging service via the above-said transport service; they can also send messages back to the server through the reverse route.

C. Device localization

As previously stated, the proposed solution is designed to trace both direct and indirect contacts. Indirect contacts tracing inherently requires device localization. Furthermore, since real-time monitoring of premises occupation is a value-added solution requirement, absolute localization of registered apps is a necessary feature, which an app-to-app Bluetooth channel simply cannot provide, and would require a dedicated Bluetooth infrastructure anyway. WiFi sensing is instead an appropriate choice to fulfil our design targets, especially if it leverages the WiFi enterprise campus infrastructure already in place.

[4]https://www.health.gov.au/resources/apps-and-tools/covidsafe-app
[5]http://tiny.cc/fljqmz
[6]https://coronamap.site
[7]http://www.koreaherald.com/view.php?ud=20200311000132
[8]https://firebase.google.com
[9]https://firebase.google.com/docs/cloud-messaging
[10]https://firebase.google.com/docs/cloud-messaging/concept-options
[11]https://spring.io
[12]https://www.heroku.com

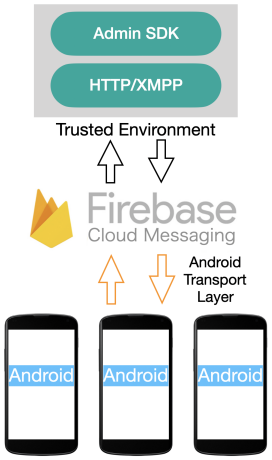

Fig. 1. Main components of the proposed system.

Notwithstanding the complex issues related to WiFi indoor localization [10], [11], our basic idea is that the AP deployed in a closed environments (e.g., a lecture hall) will turn out to be—most of the times—the one with the strongest radiating signal, thus easily providing the WiFi sensing device with absolute localization information. Experimental evidences that such a choice is compatible with most indoor areas within the campus premises can be found in the dedicated Section V.

As for outdoor spots or very noisy WiFi environments (e.g. corridors, halls), NausicaApp is provided with a feature which makes it switch the localization algorithm and talk to an external WiFi positioning system[13] through standard RESTful APIs made available by a localization provider.

In addition, NausicaApp is also triggered by a GPS-driven mechanism, which lets the device understand whether it lies within the monitored premises, where turning on the application functionalities is useful and, quite importantly, does not infringe authorization and privacy restrictions.

D. Workflow

In this section, the workflows of the above outlined application scenarios are discussed with an appropriate level of logical details. The workflow explanation follows a 4-phase scheme as outlined in Tang's paper [12] on the most outstanding solutions in the Covid-19 contact tracing arena.

Initialization phase. Each personal device running a NausicaApp instance, first, registers to FCM; in response, it gets back a registration token that uniquely and anonymously identifies it.

Sensing phase. After a NausicaApp instance has identified the AP radiating the most powerful signal strength, as discussed in Subsection III-C, it requests the Firebase SDK Admin to be registered to the relevant topic (we envisage one topic for AP's MAC/BSSID). The above request contains, besides the app instance's token, as initially registered, its *Presence Data*, which can be represented, in a loose JSON notation, as: {`AP-BSSID,Timestamp,Time-To-Live`}. The same data are temporarily stored on the local device, to be exploited for contact tracing processing as explained below.

At the server side, when the Firebase Admin SDK receives the registration request to a topic (i.e an AP's MAC) from an app instance, it checks whether the relevant app registration FCM token has not already been associated with a previous topic. If this is the case, the existing association is deleted and the new one is added. In the former case, the relevant topic counter is decremented, while in the latter it will be incremented. As soon as one of the thresholds set for each topic (i.e., each site's dominant AP) is exceeded, the interested users, namely the ones whose app is currently registered with the critical topic, are notified with a crowd gathering alert.

Reporting phase. Each time a user turns out to be infected and wishes to adhere to the containment program, her app should be first authorized (with a code released by an administrator) and then instructed to send the server the set of locally stored *Presence Data* objects combined in a *Presence History*, which we define as the temporally ordered sequence of presences stored on a device and not yet expired. The FCM back-end will then broadcast a notification with this *Presence History* as payload[14] to the rest of the anonymized and uniquely identified active devices. Any such individual (non-aggregated) information at the server side will expire within a predetermined time interval, and will be permanently erased.

Tracing phase. At client side, each device will detect suspicious contacts by matching the received *Presence History* of the infected user with its own local *Presence Data*. Temporal information derived from *Presence Data* is used to correlate direct and indirect contact tracing events. As far as presence data matching is concerned, no processing will be performed outside the personal devices.

Besides contact tracing, with its four phases, NausicaApp provides *crowd gathering monitoring*. At the server side, the logic controlling and implementing the counters associated with each FCM topic (e.g., a room's AP), or other data aggregation measures, will drive the emission of individual notifications and centralized alarms, in accordance with well-specified metrics and policies.

A screenshot demonstrating NausicaApp's operation is shown in Figure 2.

IV. Security Issues

Since the outbreak of the Covid-19 pandemic, governments, health authorities and researchers have had to face the sensitiveness of user data, with respect to deanonymization, private encounters revelation and individual movement tracking, as they tried to come up with effective contact tracing strategies, protocols and technical solutions. Special operating environments may even pose additional challenges. The focus of the NAUSICAA project is on universities, where protecting the

[13]https://en.wikipedia.org/wiki/Wi-Fi_positioning_system

[14]https://firebase.google.com/docs/cloud-messaging/concept-options#notification-messages-with-optional-data-payload

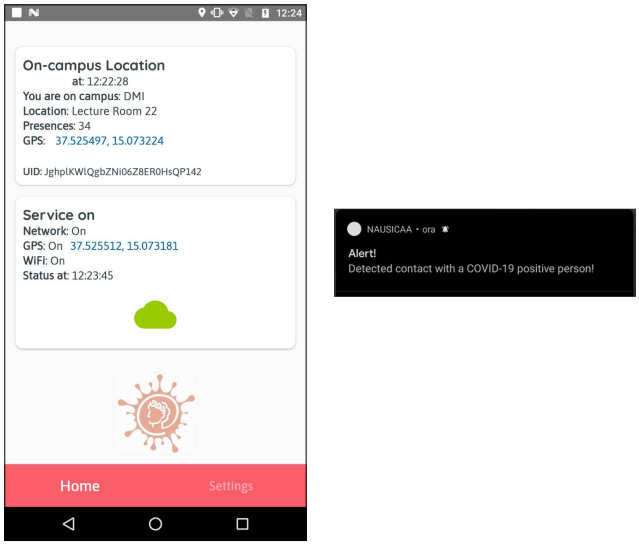

Fig. 2. Typical NausicaApp screenshot and contact alert Android notification.

rights of individuals must coexist with a steadfast effort to prevent and counteract the disruption the pandemic wreaks on the academic (teaching and research) way of life.

In the next subsections, most common vulnerabilities and attacks, such as the ones pinpointed by Vaudenay's work on DP3T proposal [5], are discussed in the context of the NausicaApp solution.

A. Channel vulnerabilities and countermeasures

The scientific community seems to have taken for granted that, to ensure privacy preservation in contact tracing initiatives based on apps running on personal devices, it suffices to pursue data origination, storing and processing following a decentralized paradigm. Quite to the contrary, many vulnerabilities in decentralized systems fail to be sufficiently addressed because of this orthodoxy. Conversely, the approach adopted in our solution, if viewed from the app-to-app channel perspective, is inherently centralized, in that it exploits combined app-to-server and server-to-app channels supported by WiFi infrastructure, and lacks a direct app-to-app channel, which is typically based on Bluetooth in most solutions.

To start with, the absence of a direct Bluetooth channel implies that an adversary cannot passively obtain pieces of personal information from monitoring (extended) Bluetooth beacon broadcast, such as MAC addresses (when transmitted), explicit user IDs or ephemeral IDs that can jeopardize anonymity. However, in our system, crucial attention is paid to ensure the deepest protection in the WiFi channel carrying comparable information from apps to apps. Since our solution can exploit the University's enterprise WiFi infrastructure, such confidentiality targets can be achieved at the minimal cost of leveraging the well-managed message encryption algorithms protecting the WiFi transmission.

At the application level, in a typical HTTPS framework, the security measures related to servers' authenticity and trustfulness are always in place, thus guaranteeing apps that the infected people's *Presence Data*: {`AP-BSSID, Timestamp, Time-To-Live`} received by the server are authentic. In particular, in order to prevent malicious sending of critical information by an attacker, such as false infected people's presence data, an additional authorization scheme is needed to upload such data to the authority side. The current solution adopts an external authorization mechanism, built in the app, whereby inputting sensitive information to the system is allowed only by a central authority that releases a unique code following the check of forwarded medical documentation from the user.

As observed in DP3T's documentation [4], data encryption is not strictly necessary at server side, since users' data privacy depends on the very nature of the transmitted data. In our solution, for instance, NausicaApp's data anonymization scheme, achieved by means of the FCM tokens, suffices to preserve users' data privacy.

B. Active and passive attacks

When it comes to a large variety of active and passive attacks that can be brought to a typical DP3T-like solution, it has to be said that NausicaApp's architecture itself, by its lack of a direct Bluetooth app-to-app channel, is capable of preventing many of them. For instance, those falling into the category of "False Alert Injection Attacks" [5], such as Backend Impersonation, False Report, Reply and Relay attacks, have few chances to succeed, since they are based on the exploitation of the Bluetooth proximity communication at a certain point in time of the attack strategy.

However, threats posed by "collected data linkability risks" [12], cannot be avoided just by stopping injection attacks, because correlations can be inferred in specific circumstances even with limited data sets (e.g., only two users are present in the same spot at a coincident time). The control logic coded in the app counteracts this risk by making decisions on how and when to show the complete set of information relayed by the server to the app user.

V. Device Localization Tests

In this section a few outcomes of the experimental tests carried out on our solution prototype are given and discussed, which have underpinned our design decision to use the WiFi campus infrastructure as a reliable way to localize the enabled devices. For campus indoor premises, the driving idea is indeed that, in any meaningful spot (e.g., classroom or laboratory), an internal *dominant* AP radiates the most powerful RSSI, thus allowing each user app to easily identify it as the one to be included in the *Presence Data* object in the {`AP-BSSID`} field. In order to show such an evidence, a few trials have been carried out with six different NausicaApp-enabled Android devices moving within and around classrooms. Each device broadcasts four WiFi scan requests (about) every two minutes, and sends the collected data, a scan quadruple in JSON notation, to the Firebase server for further processing.

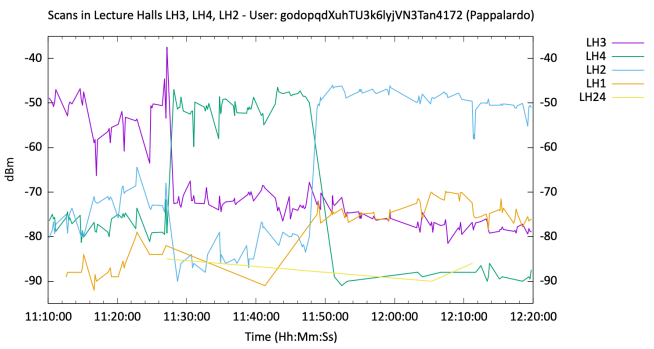

Fig. 3. RSSI vs. time for a test user moving across three lecture halls.

The experimental setup on a testing *client machine* consists of a variety of (python, awk, sed, sh) scripts, whereby scan data are collected by a per-user Firebase *listener* as the server stores them, suitably processed by a set of filters and fed to gnuplot for real-time visualization. Figure 3 plots RSSI signal strength in dBm against time over a suitable interval within which the user monitored by the listener moves between (and within) adjacent lecture halls LH3, LH4 and LH2. It is easy to see that in each hall the internal AP is easily detected as the source of the strongest received RSSI (note how crossing points, where the highest signal changes, correspond to the user changing hall). These and many other converging experimental results confirm that it is sensible to implement a maximum-signal logic within NausicaApp to identify the "local" AP's MAC to be included in the *Presence Data*.

VI. Conclusions

This paper has proposed NausicaApp, an approach for tracing contacts among users within a University campus (or comparable communities, e.g., corporate sites, schools, hospitals, etc.).

NausicaApp improves on existing approaches, as it goes beyond tracing basic *direct* person-to-person contacts, by also revealing *indirect* ones, i.e., that users happened to be in the same spot within a chosen time interval. Moreover, NausicaApp adopts stricter privacy and security measures. User privacy is ensured by avoiding to store at server side tokens identifying apps running on personal mobile devices.

A further advantage of NausicaApp is *self-localization* by user devices, obtained by having them sense signal strength radiating from WiFi Access Points (APs). WiFi scan-based positioning has indeed proved effective, simple to implement and convenient, in the presence of a majority of campus sites served by a dominant AP. Administrative access to the infrastructure is not even required, although it could ease (but is definitely not essential to) the task of mapping APs' BSSIDs to meaningful names (e.g., `78:72:5d:69:91:40` should be identified as *Library Hall* to users). Given the relatively low cost of adding some APs to a WiFi campus infrastructure, administrators might even consider to provide every significant area or site with a dominant AP, so that this may act as a "beacon" for the benefit of NausicaApp.

Admittedly, a weakness of WiFi positioning is that it is currently unavailable (or quite cumbersome to implement) for iOS devices. This actually results not from a technological limitation of the device, but a policy adopted by Apple, who do not offer WiFi scan functions in their iOS APIs.

Smooth integration of blockchain features into the current system is under design [15], with a view to distributing anonymized or aggregated data to interested parties. This will enable crowd gathering monitoring and epidemiological surveillance to be carried out in such a way that decentralization, trustworthiness and transparency are strongly enhanced [13], [14].

Acknowledgments

The authors acknowledge the support provided by project TEAMS—TEchniques to support the Analysis of big data in Medicine, energy and Structures—Piano di incentivi per la ricerca di Ateneo 2020/2022.

References

[1] M. Zastrow. South Korea is reporting intimate details of COVID-19 cases: has it helped? *Nature*, 2020.

[2] H. Cho, D. Ippolito, and Y.W. Yu. Contact tracing mobile apps for COVID-19: Privacy considerations and related trade-offs. *arXiv preprint arXiv:2003.11511*, 2020.

[3] Apple Inc. & Google LLC. Exposure notifications: Using technology to help public health authorities fight COVID-19. Web page. URL: https://www.google.com/covid19/exposurenotifications, 2020.

[4] C. Troncoso et al. Decentralized privacy-preserving proximity tracing. *arXiv preprint arXiv:2005.12273*, 2020.

[5] S. Vaudenay. Analysis of DP3T. *IACR Cryptol. ePrint Arch.*, 2020:399, 2020.

[6] G. Kwon, J. Kim, J. Noh, and S. Cho. Bluetooth low energy security vulnerability and improvement method. In *Proc. of IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pages 1–4, 2016.

[7] T. Altuwaiyan, M. Hadian, and X. Liang. EPIC: efficient privacy-preserving contact tracing for infection detection. In *Proc. of IEEE International Conference on Communications (ICC)*, pages 1–6, 2018.

[8] A. Trivedi, C. Zakaria, R. Balan, and P. Shenoy. WiFiTrace: Network-based contact tracing for infectious diseases using passive WiFi sensing. *arXiv preprint arXiv:2005.12045*, 2020.

[9] J. Bay, J. Kek, A. Tan, C. S. Hau, L. Yongquan, J. Tan, and T. A. Quy. BlueTrace: A privacy-preserving protocol for community-driven contact tracing across borders. *Government Technology Agency-Singapore, Tech. Rep*, 2020.

[10] D. Jaisinghani, R. Balan, V. Naik, A. Misra, and Y. Lee. Experiences & challenges with server-side wifi indoor localization using existing infrastructure. In *Proc. of EAI Intern. Conf. on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 226–235, 2018.

[11] G. Verga, A. Fornaia, S. Calcagno, and E. Tramontana. Yet Another Way to Unknowingly Gather People Coordinates and Its Countermeasures. In *Proc. of Internet and Distributed Computing Systems (IDCS)*, pages 130–139, Springer LNCS 11874, 2019.

[12] Q. Tang. Privacy-preserving contact tracing: current solutions and open questions. *arXiv preprint arXiv:2004.06818*, 2020.

[13] S. Micali. Algorand's approach to COVID-19 tracing. https://www.algorand.com/resources/blog/silvio-micali-approach-to-covid-19, 2020.

[14] A. Khurshid. Applying blockchain technology to address the crisis of trust during the COVID-19 pandemic. *JMIR Medical Informatics*, 8(9):e20477, 2020.

[15] G. Marotta, A. Fornaia, A. Moschitta, G. Pappalardo and E. Tramontana. NausiChain: a mobile decentralised app ensuring service continuity to university life in Covid-19 emergency times. *ICSIM2021 The 4th International Conference on Software Engineering and Information Management*, January 16-18, 2021, Yokohama, Japan.