Analyzing the Usage Behavior regarding the Corona-Warn-App

Simon Brehm Martin Heyden Lukas Marckmiller Niklas Schwabe

March 18, 2021

Abstract

The Exposure Notification Framework (ENF) designed in a joint effort by Apple and Google as well as the German Corona-Warn-App (CWA) utilizing its services aim to provide a digital solution supporting the tracing of SARS-CoV-2 infection chains. In our work, we deploy Bluetooth sensors to collect notifications emitted by active CWAs in range in order to analyze the application's spread and usage among citizens. To consider different usage scenarios, we perform scans of different types of locations like supermarkets and public parks. Moreover, we combine some scans with observations of the scanned area to improve the accuracy of our results. We relate data published by the Robert Koch-Institut to our measurements, attempting to estimate the application's effectiveness in achieving its objective to interrupt infection chains.

1 Introduction

In late December of 2019, the novel coronavirus SARS-CoV-2 was discovered in Wuhan, China, subsequently spreading worldwide rapidly, with millions of people contracting the virus. Due to its severity, the World Health Organization (WHO) classified the situation surrounding the respiratory disease, which was named COVID-19 in the process, as pandemic in March 2020. With airborne transmission being the main transmission route for SARS-CoV-2, it is key to perform social distancing whenever possible to prevent the spread of infections among one another. In scenarios where a sufficient physical distance between people cannot be ensured, maintaining a list of contacts can support the tracing of infection chains and its interruption by imposing quarantine on close contacts of an infected person. Therefore, governments across the globe soon pushed towards digital solutions aiming to support the containment of the virus' spread. In many cases, smartphone applications have been developed as a tool to automatically trace a user's contacts and to notify its user in case of a contact with a person carrying the virus.

In Germany, the Corona-Warn-App (CWA) was introduced in June 2020, utilizing the Exposure Notification Framework (ENF) provided by Apple and Google [10]. As of March 2021, the Corona-Warn-App has been installed on over 26 million mobile devices [13]. Due to its decentralized design, it is cumbersome to obtain reliable data on the usage behavior of CWA users and the application's effectiveness in confining the virus' spread. CWAs installed on mobile devices are considered active, if the underlying ENF is running, hence communicating with other active frameworks in range by utilizing the device's Bluetooth interface. For this purpose, the user is required to grant the CWA access to the ENF.

In this report, we examine the usage of the Corona-Warn-App among citizens in different types of locations, relating the number of CWA downloads as published by the Robert Koch-Institut [13] to actually active CWAs as observed in our experiments. Further, an active CWA is particularly valuable if citizens gather in groups. Therefore, we investigate whether people moving in groups tend to carry devices that run active CWAs. In our analysis, we emphasize the handling of infections reported within the Corona-Warn-App. We match the data collected in our experiments to the data published by the CWA system and the Robert Koch-Institut to estimate the effectiveness of the Corona-Warn-App regarding its purpose of contributing to confine the virus' spread.

2 Preliminaries

The German Corona-Warn-App builds atop the Exposure Notification Framework, which was developed and integrated by Apple and Google in their mobile operating systems iOS and Android respectively in a joint effort. It aims to record any encounters with other people and warns its user about the risk of an infection due to the exposure to citizens that later turned out to carry the SARS-CoV-2 virus. For each encounter, the duration of the meeting and the distance to every participant are included to calculate an individual risk score, determining the type of warning a user receives. In the following, we distinguish the tasks of both the ENF and the CWA system.

2.1 The Exposure Notification Framework (ENF)

The Exposure Notification Framework randomly generates a private 16-byte *Temporary Exposure Key* (TEK) once a day, storing the most recent 14 TEKs locally in the device's storage. Within a day, 16-byte *Rolling Proximity Identifiers* (RPI) are derived from the current TEK and an increasing interval number once about every 10 minutes, thereby generating up to 144 distinct RPIs per day. Each TEK is associated with an initial interval number, which is determined by dividing the Unix Epoch Time at TEK generation by 600 seconds. Given an RPI, it is neither possible to determine its corresponding TEK nor to correlate other RPIs to that particular RPI [11]. Every active ENF notifies its surroundings about its presence by broadcasting Bluetooth Low Energy (BLE) advertising packets, which can be identified as Exposure Notifications by its service UUID `0xFD6F`. Its payload comprises the current RPI and 4 bytes of *Associated Encrypted Metadata* (AEM), which indicates the currently used framework version and the sender's transmission power while emitting the advertisement. To decrypt the AEM, the corresponding RPI and an AEM key are required, which is derived from the respective TEK as well [11]. Figure 1 depicts the entire 31-byte advertisement payload structure as defined in the specification provided by Google and Apple [10]. However, in practice the leading 3-byte section containing a set of flags is omitted in some cases (cf. Section 6).

Besides emitting messages as stated, an ENF also scans for BLE advertisements periodically to detect other active ENFs in range. Thereby, considering the sender's transmission power, signal strength and the duration of the contact, an ENF can estimate the severity of an exposure to the person carrying the device that emitted the respective notifications. The RPIs propagated in the received advertisements are stored locally on the mobile device. Note that multi-byte values in BLE packets are given in little-endian order.

Flags			**Complete 16-bit Service UUID**			**Service Data – 16-bit UUID**				
Length	Type	Flags	Length	Type	Service UUID	Length	Type	Service Data		
`0x02`	`0x01`	`0x1A`	`0x03`	`0x03`	`0xFD6F`	`0x17`	`0x16`	`0xFD6F`	**16 bytes**	**4 bytes**
	(Flag)			(Service UUID)	(Exposure Notification Service)		(Service Data – UUID)	(Exposure Notification Service)	**Rolling Proximity Identifier**	**Associated Encrypted Metadata**

Figure 1: Payload format of BLE advertisement emitted by the ENF [10].

To evaluate possible exposures, the CWA commits a list of TEKs to the ENF that it obtained from the CWA server periodically. Similarly, the ENF provides the 14 most recent TEKs to the CWA if a user reports an infection within the application and consents to share its diagnosis with other users. For every TEK received from the CWA, the ENF can derive all corresponding RPIs to determine matches with the RPIs stored on the device. If an exposure is detected due to a matching RPI, the calculated risk score regarding that exposure is fed back to the CWA, which displays a warning to its user.

2.2 The Corona-Warn-App (CWA) system

The German CWA system developed by SAP and Deutsche Telekom is composed of two main components [15]. A mobile application provides a graphical user interface that allows its user to determine its current exposure risk level based on the calculation by the ENF. Users may enable or disable the functionality offered by the framework, i.e. broadcasting BLE advertisements containing one's current RPI and receiving RPIs from other active ENFs nearby. Occasionally, new features are incorporated into the app, for instance, statistics on the number of new infections within the past 24 hours and the number of reported cases within the app are displayed since the roll-out of an update in late January.

To obtain that figures, the mobile application periodically polls the system's second component, the CWA server. Besides this auxiliary information, the server also provides lists of all TEKs that were uploaded by infected individuals within the past 14 days. To share one's keys, the respective user has to provide consent within the mobile application. The thereby obtained list of TEKs is then passed to the ENF, which determines the risk of exposure utilizing the TEKs as described in Section 2.1.

3 Solution approach

In order to analyze the usage behavior of CWA users, access to the RPIs emitted by a user's mobile device via BLE is required. Mobile sensors equipped with a Bluetooth module can be programmed to act as a scanner, collecting BLE beacons in range, particularly any beacons emitted by an active Exposure Notification Framework. Beyond that, the interface provided by the CWA server allows to gather the Temporary Exposure Keys uploaded by infected CWA users. Deriving the corresponding RPIs for each TEK, one is capable of detecting infections among the people whose RPIs have been recorded by a sensor.

While placing sensors in crowded locations is particularly valuable to collect large amounts on RPIs, it is cumbersome to derive reliable statements about the CWA usage by the recorded RPIs alone. Therefore, we also conduct experiments to match individuals against emitted ENF packets. Despite the employed techniques aiming to preserve a user's privacy while running a CWA, this correlation can be accomplished by observing a sensor's surroundings. As long as the area around a sensor is sparsely populated, the occurrence of an ENF advertisement may be attributed to an individual approaching the sensor at the same time. Given that information, we estimate the fraction of citizens that carries a device running an active CWA, considering different types of locations. Moreover, we explicitly focus on the fraction of CWA users for groups of people passing a sensor. In this scenario, the CWA could be particularly helpful to trace contacts after an infection, however, its effectiveness is highly dependent on the CWA usage among the group members.

4 Implementation

In order to analyze the users' behavior regarding the Corona-Warn-App, access to the data provided by the Exposure Notification Framework is required. Further, CWAs communicate with the CWA server periodically to collect recent information about infected users and statistics about infection events in Germany provided by the Robert Koch-Institut. Our implementation mimics relevant parts of the ENF and the Corona-Warn-App, since a large fraction of the desired data is not accessible to an ordinary user. Utilizing Bluetooth Low Energy sensors, we scan the sensor's surroundings to collect any beacon emitted by an active ENF nearby. The sensor's implementation is detailed in Section 4.1. Beyond that, we fetch any information that the CWA server provides about occurring infections on a daily basis. In a final step, the beacons gathered by the sensors and the RPIs derived from the data provided by the CWA server can be compared to detect infections among CWA users who passed a scanning sensor in close proximity. Moreover, one may be able to draw further conclusions about the CWA and its users' behavior by correlating the datasets with additional information, for instance physical observations of a sensor's surroundings. In Section 4.2, we thoroughly describe the implementation of the tools we employ to process and subsequently analyze the collected data. Our implementation and all datasets gathered during our experiments are accessible on GitHub [2].

4.1 ESP32 sensor programming

We utilize C++-programmed ESP32 sensors, namely the *Adafruit* HUZZAH32, to scan various areas, collecting any BLE beacons in range and filtering out Exposure Notification packets. BLE advertisements commonly contain a UUID that indicates the service the advertising device offers. The Exposure Notification service was registered with `0xFD6F` as its service UUID, so filtering the collected beacons by that UUID is straightforward. Each advertisement carries a single 16-byte Rolling Proximity Identifier and 4 bytes of Associated Encrypted Metadata as part of its payload. Beyond that, we record further auxiliary information about the beacon, which is detailed in the following.

The *Espressif IoT Development Framework* (ESP-IDF) [4] offers two fundamentally different options to obtain temporal information. On the one hand, one may synchronize a sensor using the *Simple Network Time Protocol* (SNTP) [7]. On the other hand, the built-in function `esp_timer_get_time()` provides the time spent since the sensor's start-up utilizing a high-resolution hardware timer [5]. Since a reliable connection to the Internet cannot be guaranteed in most locations where the sensors are placed, the first option is infeasible in our use case. By manually synchronizing a sensor's start-up with an external clock, we are able to obtain a timestamp with sufficient precision using the hardware timer to determine the moment in which a beacon started recording in retrospect.

During our experiments we noticed that some BLE advertisements emitted by the ENF deviate from the structure documented in the specification [10]. Some devices omit the first 3 bytes of the BLE packet which contain a set of flags, thus shrinking the packet size from 31 bytes to 28 bytes. To estimate the commonness of this deviation while efficiently utilizing the limited amount of storage, we allocate two additional bits to store information about the beacon's length and structure without the need to retain the entire packet. To assess the distance of an individual to a sensor, we are further capable of recording the *Received Signal Strength Indication* (RSSI) of a perceived BLE packet.

For each beacon, the RPI and its corresponding AEM are stored in the sensor's non-volatile memory coupled with the timestamp of the beacon's occurrence and two bits that indicate the packet structure as describe above. Depending on the use case, the RSSI is stored as well. Since the smallest unit to store data is a single byte, we opted to store the timestamp using 22 bits, assigning the remaining two most significant bits to the packet structure indication. The timestamp thereby may represent up to 2^{22} seconds since start-up, which in theory allows the sensor to run for over 48 days without overflowing.

To ensure persistence of the collected data even if the connected battery depletes, we utilize the SPIFFS library [3], which by default allows to access about 1.4MB of an ESP32 sensor's non-volatile storage. This enables the storage of more than 60,000 RPIs and information associated with it. Since SPIFFS writes pages of 256 bytes, we implemented a buffer using the C++-built-in `vector` as a data structure, which allows to efficiently append data to its end while being dynamically resizable to cope with different quantities of beacons within a scan interval. Instead of writing to a sensor's flash storage immediately after perceiving a BLE advertisement, the relevant data is kept in-memory until the specified scan interval has passed. Before going to sleep, the data accumulated in the buffer is written to non-volatile storage only executing a single write operation, thereby increasing the flash cells' lifetime. Since write operations to flash storage usually take much longer than keeping data in-memory, buffering the collected data also contributes to ensuring a smooth operation while scanning for BLE advertisements.

We experimented with a duplicate detection mechanism, comparing RPIs kept in the buffer to perceived advertisements in order to avoid storing RPIs repeatedly. However, we decided to dismiss that idea considering the following: Firstly, our experiments were constrained rather by battery life than by storage size. Comparing each recorded beacon to potentially dozens of buffered beacons is expensive, thus draining the battery faster. Secondly, being aware of multiple occurrences of the same RPI is valuable for later analysis, for instance when determining the duration a user resides in close proximity to the sensor.

After a scan interval is completed and the collected data has been stored in non-volatile memory, the sensor is sent to a built-in sleep mode for a specified number of seconds. In this light sleep mode, the sensor powers down several of its components, including the BLE chip and its I/O interface [6]. Utilizing the sleep mode significantly increases the runtime of the sensor due to the energy saved while residing in this state. The programming interface also provides a deep sleep mode, however, that mode is impractical for our use case, since it resets the timestamp we refer to when storing collected RPIs.

Once we have fetched the distributed sensors, we access the sensors' flash memory to transfer the gathered data to more powerful machines for later analysis. For safety reason, we decided to outsource the clearing of the sensor's flash storage to a separate program. The code used to scan for ENF packets exclusively allows to append to the data stored in flash storage, thereby, the accidental deletion of any collected data is ruled out by design.

4.2 Evaluation Framework

Since ENFs only exchange RPIs among another, we need to mimic the CWA ecosystem to determine whether our sensors collected BLE advertisements that carry an RPI of an infected person. The ecosystem comprises a server and a client. The Corona-Warn-App serves as a client, exchanging RPIs with other CWAs in range. Our sensors mimic these clients by collecting RPIs emitted by other devices. To conclude that a collected RPI was transmitted by an ENF running on the mobile device of an infected user, its corresponding TEK is needed. Each day, the server in the CWA ecosystem provides the TEKs of new infected users by publishing a list of TEK structures of the preceding day.

A TEK structure consists of six attributes and needs to decoded before use. For our experiments we only require a subset of the provided attributes: The TEK itself (`key_data`) allows to derive its corresponding RPIs and the AEM key used to decrypt the AEM related to an RPI. The `rolling_start_interval_number` determines the first interval number that can be used to derive valid RPIs utilizing the associated `key_data` and is equal to the initial interval number described in Section 2.1. Finally, the `rolling_period` defines the validity period of a TEK in multiples of 10 minutes. One can determine all interval numbers that yield valid RPIs for the corresponding TEK by incrementing the `rolling_start_interval_number` until the sum of the `rolling_start_interval_number` and the `rolling_period` is reached.

As part of our evaluation framework, we implemented a component that mimics a CWA client by polling the server every day to obtain its list of TEK structures, subsequently decoding and storing it for further use. The server provides the data in a serialized format using *Protocol Buffers*. To decode the serialized data, we created a decoder utilizing the Protocol Buffers Compiler *protoc* [8] and the *Proto* files provided by the CWA GitHub repository for iOS [14]. At the end of January, the CWA ecosystem was updated, with the server now providing additional statistics like the amount of daily infections throughout Germany and the number of warnings by CWA users. We updated our implementation accordingly to include these statistics in the polling process and updated our Protocol Buffers decoder too, since the statistics are provided in serialized form as well.

Another framework component treats every TEK that was previously stored by our polling component. For every TEK, we derive up to N RPIs according to the ENF cryptography specification [11], where N represents the `rolling_period` contained in the corresponding TEK structure. Each derived RPI is matched with the RPIs that we collected with our sensors. If a match occurs, we can conclude that a person that passed one of our sensors uploaded its TEKs to the server after testing positive for SARS-CoV-2. Furthermore, for each derived RPI that matched with our collected RPIs, the AEM is decrypted to access the sender's BLE transmission power. To test our implementation regarding the cryptography operations, we utilized data provided by Baumgärtner et al. [1]. Further, the component analyzes the statistics provided by the server and iterates over all collected beacons to output data which is relevant for our analysis in Section 6, such as the commonness of the deviation from the ENF Bluetooth specification. The remaining components of our framework enable a fine-tuned, automatic evaluation of different experiments which are detailed in Section 5.

5 Experiments

Given the implementation described in Section 4, we utilized the programmed sensors in several types of experiments. To argue about the collected data, it is key to estimate the radius a sensor covers when scanning for BLE beacons in the first place. With that, one can determine suited locations to place a sensor, for instance spots that many people have to pass due to a physical bottleneck. Further, being well acquainted with the sensor's behavior may yield more precise results when physically observing the area around a sensor. The assessment whether a person came sufficiently close to be perceived by the sensor proved to be a main challenge conducting the respective experiments. While a reliable detection also depends distinctly on the sender's transmission power, we estimate that the deployed sensors reliably detect beacons emitted by an ENF within a distance of five meters as long as no major obstacles are present.

Collecting large amounts of RPIs to detect infections. The experiments we conducted can be clustered in two main categories, pursuing different objectives respectively. To begin with, we placed sensors in several well-frequented locations to collect as many RPIs as possible. Due to the continuous shutdown in Germany during the course of our studies, we opted to contact different facilities excluded from the restrictions imposed by the government. Hence, we were able to scan for active CWAs in several grocery stores and supermarkets in Darmstadt, and even were permitted to place two sensors in a hospital in Munich. Often it was possible to situate the sensors in close proximity to either the entrance or the checkout, both being bottlenecks that most customers pass at least once during their visit. Additionally, the sensors are less prone to theft when being placed inside these facilities. In the process, we tweaked each sensor's scan and sleep duration, adjusting the parameters based on the respective location. Given that customers often reside at the checkout for a while, a pause of around 30 seconds between two scan intervals appears to be appropriate to save battery power and keep the probability of missing out on beacons low. Beyond that, we occasionally carried a sensor instead of a smartphone that runs the Corona-Warn-App when visiting localities with many people, for instance when riding a bicycle on trafficked roads or while traveling by public transport.

Observing a sensor's surroundings to determine CWA usage. With the second portion of our experiments, we aim to give an estimate of the fraction of citizens that actively uses the Corona-Warn-App, comparing it to the officially published information by the Robert Koch-Institut [13]. Further, it is also worth investigating whether the CWA is used when gathering in groups, since the tracing of infections is particularly useful in that case. To estimate the spread and the effectiveness of the Corona-Warn-App in that scenario, we combine the data collected by a sensor with physical observations of the sensor's surroundings. Depending on the quantity of people nearby, we either merely count the number of people passing the sensor with a sufficiently small distance, or we additionally record a timestamp whenever a person approaches the sensor. Thereby, it is feasible to attribute recorded RPIs to small groups of citizens or even to single individuals.

To obtain reliable data, we impose several requirements on the sensor's location when performing the measurements as describe above. First, passing the sensor should be inevitable for anyone reasonably close in order to avoid that people move on the edge of the sensor's range. Being approachable from only two directions further eases the enumeration of people nearby. Also, a sensor should be placed in sufficient distance from any roads frequented by cars, since it is challenging to estimate whether a sensor reliably perceives beacons emitted by passengers moving past the sensor with significant speed. Considering these restrictions, we gathered Exposure Notification packets coupled with observations of the area at the main railroad station of Darmstadt, public parks and the city's periphery.

Arising problems while conducting the experiments. Unfortunately, lots of stores that we contacted were reluctant to place a sensor in their facilities, some having reservations regarding privacy and the Corona-Warn-App itself. From a technical perspective, the sensors' energy consumption was a significant issue in the beginning, only running for a single day in most cases. However, through continuous improvements of the sensor's code and its parameters regarding the scan and sleep duration, we were able to extend the sensors' runtime to almost five days by the end of our studies. Yet, some ESP32 sensors showed erratic behavior, occasionally resetting during operation or corrupting the file used to store the collected beacons. In the latter case, formatting the file system was the only solution to return to normal operation. Moreover, one sensor even caught fire after being connected to a computer's USB port. During the experiments itself, gauging the sensor's range during operation emerged as a major challenge.

6 Evaluation

During our experiments, we were able to collect **97,561** unique RPIs between mid-December of 2020 and the beginning of March 2021. For **17** TEKs uploaded to the CWA server as the result of a positive SARS-CoV-2 test, we recorded at least one associated RPI. In total, **31** matches occurred among the gathered RPIs, hence, some individuals' RPI rotated while being in range of one of the deployed sensors. While the number of traced users who contracted the SARS-CoV-2 virus and reported their test result within the CWA is too small to designate spatial hotspots where the risk of an infection is high, we still compare our measurements to the figures published by the Robert Koch-Institut (RKI) on the occurrence of infections. For Darmstadt, the RKI reported approximately 1,500 infected citizens in the experiment period [12].

Figure 2 depicts the daily number of infections in Germany being reported to the RKI since the start of our project. Further, for each day, we determine the number of reports within the CWA by interpreting the TEKs obtained from the CWA server for that particular day. Each uploaded TEK is coupled with a timestamp that indicates its day of generation. By enumerating all TEKs that were generated on the day prior to the upload, one obtains the number of reports within the application. Utilizing the daily statistics that are published within the CWA since the end of January, we are able to validate our results. With only **13%** of the infections being reported in the CWA and roughly **59%** of the CWA users sharing a positive test result [13], it becomes evident that the application's effectiveness in supporting the tracing of infections is significantly diminished. Therefore, enabling the reception of a test result within the application from all testing facilities and insistently motivating its users to share an infection is indispensable.

Detailing the discovery of infected users. For most TEKs that were reported to the CWA server, our sensors did not record any encounters with other CWA users who shared a SARS-CoV-2 infection via the application. In many cases, the associated RPIs only occurred in sensor range for a short period. However, at the *Rewe Center* at Leyndecker

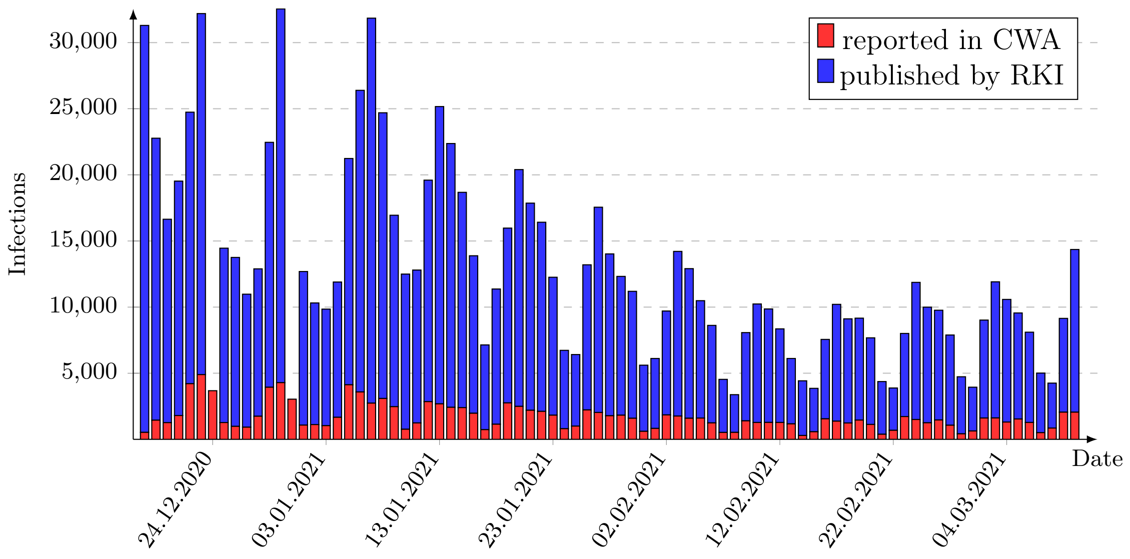

Figure 2: Daily number of infections reported to the RKI, including the fraction that was shared within the CWA. No data was published for Dec 24th, 2020 and Dec 31st, 2020.

Straße 16, 64293 Darmstadt, and the *Tegut* at Kasinostraße 92, 64293 Darmstadt, our sensors recorded infected users that remained in sensor range during multiple RPI rotations. Given the timestamps that were stored in association with the RPIs, we can determine a lower boundary for the infected users' duration of stay. For instance, at the Rewe Center, an infected individual was recorded by the deployed sensor between 15:46 and 16:21 on February 11th, 2021, with its device emitting four subsequent RPIs. Conducting an experiment in Langen, a set of RPIs corresponding to a single TEK was recorded for the entire duration of the experiment, which we conducted for about an hour. We suspect that a vendor nearby contracted the virus and shared its infection in the CWA subsequently.

In three cases, we observed a particularly interesting scenario, where two different individuals that reported their infection within the CWA were logged by a sensor at around the same time. At the Rewe Center, the respective RPIs were recorded only few times with quite some temporal distance, impeding any analysis. At the Bürgerpark Nord, the stored RPIs indicate that both infected individuals walked through the park together, however, given the small set of RPIs, this assumption is prone for error. In a third case recorded at Tegut, the analyzed dataset is larger, yielding more expressive data. On January 16th, 2021, one infected individual was traced by its RPIs from 13:32 until 14:08, while the sensor recorded the second infected individual between 13:56 and 14:27. Unfortunately, the sensor's battery depleted at that point. Since the sensor was placed close to the supermarket's entry, the first occurrence of a respective RPI likely corresponds to the entering of the market. For both TEKs, the corresponding RPIs were recorded in almost every sensor scan interval. Given the distance of time between the first occurrence, we assume that the infected users did not enter the market together. However, the signal strengths that the sensor recorded for the advertisements perceived from both ENFs indicate that both individuals came close to the sensor for several minutes at the same time between 14:00 and 14:05. The stronger two distinct signals are when perceived by the sensor, the closer the broadcasting devices must have been to each other as well. Given that circumstances, we may have recorded an event where a transmission of the SARS-CoV-2 virus took place.

Determining periods with high customer activity. For the aforementioned Rewe Center and Tegut as well the *Unverpackt Darmstadt* at Gutenbergstraße 5B, 64289 Darmstadt, we analyzed the frequency of recorded RPIs during the stores' opening hours. Thereby we can provide an estimate on the amount of customers throughout the day, determining periods at which the infection risk may be reduced due to a lower number of customers present. Our results are depicted in Figure 3. To obtain valid data, we only consider recordings from sensors that were running for a market's entire opening time and group the collected RPIs in one-hour intervals. Taking account of different amounts of customers among the markets, we normalize the number of RPIs. A one-hour period with a value of 1 constitutes the highest load of customers measured by the deployed sensor. All other values are aligned to the maximal amount accordingly.

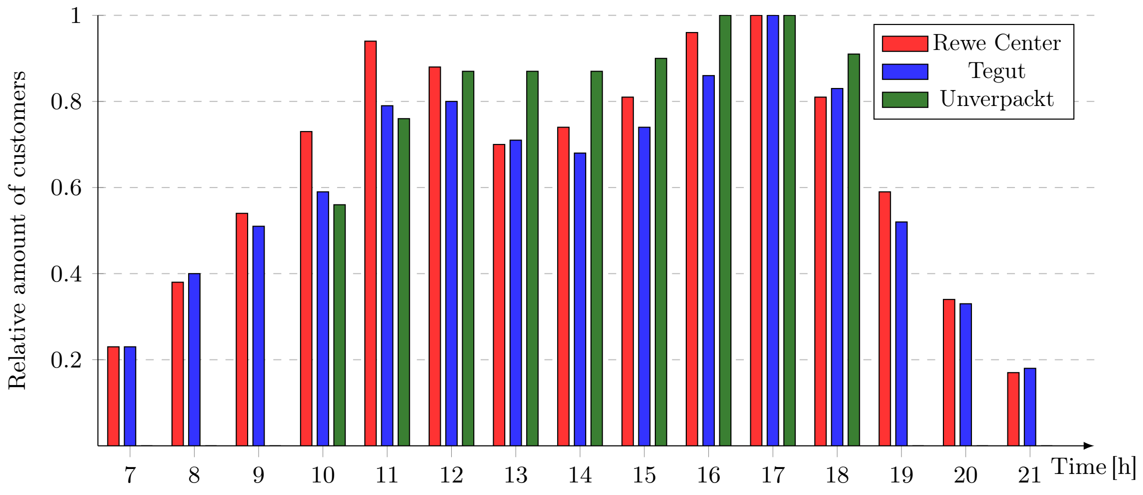

Figure 3: Distribution of the amount of customers during the markets' opening hours. Unverpackt Darmstadt has divergent opening hours (10:00 – 19:00).

It is evident that the time approaching noon and the late afternoon around 17:00 are peak hours, at which lots of customers equipped with a CWA are present at the markets. Clearly, the stores are less populated in the first hours after opening and towards the evening, with a slight drop in the number of customers at lunchtime. Avoiding the stated peak hours when shopping, one may reduce the risk to expose oneself to an infected individual. While our sensors naturally only record CWA users, we presume that our results are comparable to the actual distribution of customers throughout the day.

Considering deviations from the specification. As presented in Section 2.1 and stated in the specification published by Google and Apple [10], the BLE advertisement payload broadcasted by an active Exposure Notification Framework is composed of three sections. With a total length of 31 bytes, the first 3-byte section comprises a set of flags, indicating the sender's transmission mode. The second section determines the service that the packet originated from using 4 bytes, with the remaining bytes carrying an RPI and its respective AEM. During our measurements we occasionally perceived beacons that lacked the latter section, thus not carrying any RPI at all. Since this observation only affects about **1.1%** of the recorded beacons, we do not examine these cases in detail.

However, a significant number of devices running an ENF omits the leading 3-byte flag section when broadcasting BLE advertisements. We recorded information on the packet structure for a subset of **51,693** RPIs, with **25.5%** of all distinct ENF advertisements showing this kind of behavior. Correlating BLE advertisements to a number of our private devices, namely a Fairphone 3, OnePlus 5T, OnePlus 8 Pro, Samsung Galaxy S5/S8/S9+, Samsung Galaxy Note 10+ and a Huawei Mate 20 Lite all running Android, and the iPhone 8 and iPhone SE 2016 running iOS respectively, we hypothesize that this deviation is indeed dependent on the device's operating system. Every considered Android device broadcasted 28-byte advertisements that lacked the flag section, while all devices running iOS emitted beacons that complied with the specification.

While determining a device's operating system by its ENF advertisements is not inherently critical to security and privacy, it still poses an unnecessary disclosure of information about a user's device that could easily be avoided. More importantly, the discrepancy between the results of our measurements and the number of downloads published by the Robert Koch-Institut is noteworthy [13]. With 55% of the downloads being initiated from an Android device, the fraction of collected RPIs that were broadcasted as part of a 28-byte advertisement payload is substantially smaller with 25.5%.

As future work, experiments with a larger set of mobile devices could be conducted to attest our hypothesis. Nevertheless, we performed further experiments to determine the cause of the discrepancy between official figures and the data gathered during our experiments. Indeed, iOS devices occur to emit ENF advertisement with a significantly higher transmission power than its Android counterparts. We suspect that BLE advertisements broadcasted by the ENF that runs on Apple devices are perceived by the sensors from a higher distance due to its higher transmission power, resulting in a higher number of duplicates at the sensors. Indeed, the average number of occurrences per RPI is twice as big for 31-byte advertisements compared to beacons that lack the flag section, which supports our hypothesis regarding the attribution to a particular OS. Nevertheless, the risk calculation performed by the ENF is not affected by the transmission power variance, since it not only considers the sender's transmission power, but also includes the Received Signal Strength Indication (RSSI) in its calculations [9].

Correlating observations to the collected sensor data. With 26 millions downloads to date, approximately 31% of Germany's population has installed the CWA assuming only a single device per citizen. However, due to the application's decentralized design, information about the user behavior is sparse, for instance regarding the fraction of users that actively utilizes the CWA by granting the required permissions. We conducted the experiments at Darmstadt's *Bürgerpark Nord* a dozen times, and further performed the same observations in the periphery of Langen and the main railroad stations of Darmstadt and Mannheim. Due to the aforementioned significant variance in the senders' transmission power, we suspect that the following results are prone to error.

Overall, we observed **6,818** people passing one of our sensors, with **2,662** unique RPIs recorded in the same period, thereby yielding a fraction of **39%** that ran an active CWA. This rate appears to be reasonable despite the fact that the officially published data indicates a slightly lower distribution. However, considering individual experiments, the fraction of CWA users significantly varies, ranging from 25% up to 85% in rare occasions. For the experiments at Darmstadt central station, the rate of estimated CWA users was particularly low, with only 32% of 3,772 people being recorded by the sensor. We suspect that the sensor's position was not chosen ideally, allowing too many people to pass without being detected by the sensor. For the experiments conducted at the Bürgerpark Nord, the rate of CWA users was higher while only showing slight fluctuations, ranging between 31% and 54% in exceptional cases. The chosen location emerged to be very suitable for our experiment setup, with everyone approaching the area passing the sensor eventually within a distance of less than five meters. Finally, the observation in the periphery of Langen yielded highly varying results, with a rate of 35% to 85% using the CWA.

Extending the experiments, we put a focus on groups of people passing the sensor. While it is indeed feasible to attribute an RPI to a particular person if few people are close, the mapping emerges to be cumbersome if several (groups of) people approach the sensor in quick succession. We suppose that the accuracy of our data is too low to provide dependable statements about the CWA usage among people moving in groups. For that reason, we do not pursue a detailed analysis regarding this issue.

7 Conclusion & Future Work

We were able to collect a large quantity of beacons, successfully identifying trends in our data by applying tools for automatic analysis. For instance, we are able to determine time slots in which places like supermarkets that our sensors monitored are particularly crowded. Thereby, we can recommend periods in which an exposure to other people, particularly infected citizens, is less likely. However, the number of ascertained infections is too low to provide a general statement about spatial hotspots throughout Darmstadt. Apart from motivating citizens to download the Corona-Warn-App, it is essential to apply means that increase the number of reports of positive test results within the CWA.

Correlating recorded data to observations emerged to be cumbersome due to the nature of wireless technologies, and was further hampered by the variance in the senders' transmission power. As future work, it could be of interest to study the deviation from the standard in more detail. For that, one may utilize a wider range of mobile devices to verify our hypothesis regarding the differences in the ENF's implementation. Moreover, assessing the impact of a low transmission power on the effectiveness of the CWA is critical to ensure an operation that supports the mitigation of the SARS-CoV-2 spread.

References

[1] L. Baumgärtner, A. Dmitrienko, B. Freisleben, A. Gruler, J. Höchst, J. Kühlberg, M. Mezini, R. Mitev, M. Miettinen, A. Muhamedagic, T. D. Nguyen, A. Penning, D. F. Pustelnik, F. Roos, A. Sadeghi, M. Schwarz, and C. Uhl. Mind the GAP: security & privacy risks of contact tracing apps. In *TrustCom*, pages 458–467. IEEE, 2020.

[2] S. Brehm, M. Heyden, L. Marckmiller, and N. Schwabe. GitHub repository – Project Implementation. `https://github.com/CWAId-Catcher-Team/cwaid-catcher`, 2021. Accessed: 2021-03-13.

[3] Espressif. ESP32 SPIFFS library GitHub repository. `https://github.com/espressif/arduino-esp32/tree/master/libraries/SPIFFS`. Accessed: 2021-02-26.

[4] Espressif. IoT Development Framework. `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/index.html`. Accessed: 2021-02-26.

[5] Espressif. IoT Development Framework – High Resolution Timer. `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/esp_timer.html`. Accessed: 2021-02-26.

[6] Espressif. IoT Development Framework – Sleep Modes. `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/sleep_modes.html`. Accessed: 2021-02-26.

[7] Espressif. IoT Development Framework – System Time. `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/system_time.html`. Accessed: 2021-02-26.

[8] Google LLC. Protocol Buffers GitHub repository. `https://github.com/protocolbuffers/protobuf`. Accessed: 2021-03-13.

[9] Google LLC and Apple Inc. Exposure Notification – BLE attenuations. `https://developers.google.com/android/exposure-notifications/ble-attenuation-overview`, 2020. Accessed: 2021-03-12.

[10] Google LLC and Apple Inc. Exposure Notification – Bluetooth Specification. `https://blog.google/documents /70/Exposure_Notification_-_Bluetooth_Specification_v1.2.2.pdf`, 2020. Accessed: 2021-02-26.

[11] Google LLC and Apple Inc. Exposure Notification – Cryptography Specification. `https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ExposureNotification-CryptographySpecificationv1.2.pdf`, 2020. Accessed: 2021-03-10.

[12] Hessisches Ministerium für Soziales und Integration. Tägliche Übersicht der bestätigten SARS-CoV-2-Fälle. `https://soziales.hessen.de/gesundheit/corona-in-hessen/taegliche-uebersicht-der-bestaetigten-sars-cov-2-faelle`. Accessed: 2021-03-07.

[13] Robert Koch-Institut. Kennzahlen zur Corona-Warn-App. `https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/WarnApp/Archiv_Kennzahlen/Kennzahlen_04032021.pdf`, 2021. Accessed: 2021-03-05.

[14] SAP and Deutsche Telekom. Corona-Warn-App iOS App. `https://github.com/corona-warn-app/cwa-app-ios`, 2020. Accessed: 2021-03-13.

[15] SAP and Deutsche Telekom. Corona-Warn-App Solution Architecture. `https://github.com/corona-warn-app/cwa-documentation/blob/master/solution_architecture.md`, 2020. Accessed: 2020-12-28.