MDPI

Article

MAC Address Anonymization for Crowd Counting

Jean-François Determe [1,*], Sophia Azzagnuni [2], François Horlin [2] and Philippe De Doncker [2]

1   BEAMS-EE, Université Libre de Bruxelles, 1050 Brussels, Belgium
2   OPERA Wireless Communications Group, Université Libre de Bruxelles, 1050 Brussels, Belgium;
     sophia.azzagnuni@ulb.be (S.A.); francois.horlin@ulb.be (F.H.); philippe.dedoncker@ulb.be (P.D.D.)
*   Correspondence: jean-francois.determe@ulb.be

Abstract: Research has shown that counting WiFi packets called probe requests (PRs) implicitly provides a proxy for the number of people in an area. In this paper, we discuss a crowd counting system involving WiFi sensors detecting PRs over the air, then extracting and anonymizing their media access control (MAC) addresses using a hash-based approach. This paper discusses an anonymization procedure and shows time-synchronization inaccuracies among sensors and hashing collision rates to be low enough to prevent anonymization from interfering with counting algorithms. In particular, we derive an approximation of the collision rate of uniformly distributed identifiers, with analytical error bounds.

1. Introduction

Many an event organizer deals with crowd monitoring and management [1]. Recently, works from different teams proposed crowd counting systems using WiFi signals [2–4]. These works describe counting systems detecting special control packets of the WiFi protocol: probe requests (PRs). Such packets are periodically transmitted by WiFi user terminals to detect nearby access points. Therefore, a PR-based counting system eludes the need for user cooperation and the need for an active WiFi connection from terminals to access points within range.

Typically, several WiFi sensors are deployed over the monitored area to detect PRs and then extract and anonymize their media access control (MAC) addresses. Sensors then timestamp anonymized PRs and transmit them to a central server processing them jointly. The number of distinct PRs acquired during a time frame of T seconds (with $T = 60$ s in this paper) implicitly provides a *rate of PR transmission*, which is proportional (on average) to the number of attendees (as shown experimentally in [3] and theoretically in [5]). The proportionality between what is measured (the rate of PR transmission) and what is interesting to event organizers (the number of attendees) is referred to as the *extrapolation factor* in our previous works and is determined experimentally.

Only in circumstances where occupation varies significantly in 60 s is our system less accurate (because it averages probe requests over a time frame of one minute, thereby smoothing any occupancy change occurring over a one-minute time frame). However, both indoor and outdoor measurements in [3,5] indicate it does not seem common for events or buildings hosting at least a few hundreds of individuals, probably because they enter and leave monitored areas at different times and also because entry and exit points have limited flow capacity.

1.1. A Short Description of the Monitoring System Architecture

Figure 1 depicts the experimentally validated counting scheme in [3,4]. Sensors (three in Figure 1) monitor an area and, because their effective detection range is not known precisely (it depends on the propagation environment and decreases as the density of

people increases because of body-induced attenuation), they are usually installed densely enough to make detection ranges overlap. Data transfers between sensors and the central server are secured using hypertext transfer protocol secure (HTTPS) connections (with transport layer security (TLS)) so that the traffic is encrypted and the identity of the central server is verified—the latter preventing man-in-the-middle attacks. Sensors synchronize their clocks using network time protocol (NTP) servers.
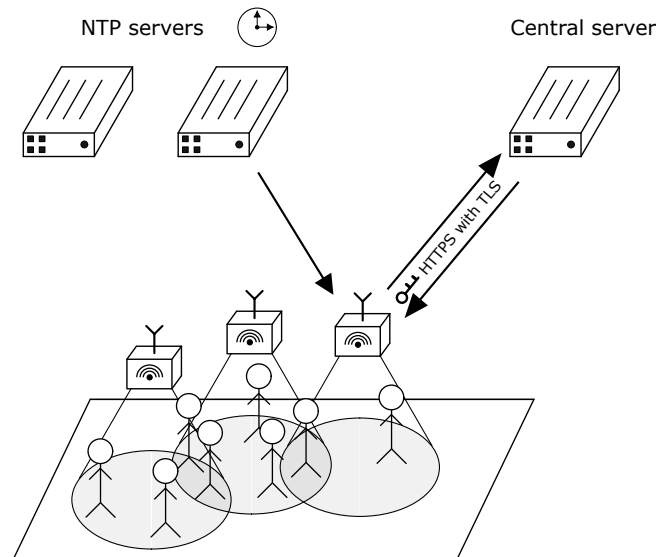

Figure 1. Scheme of the PR sensing procedure. Three WiFi sensors with overlapping ranges detect WiFi probe requests emitted by the smartphones of individuals. The shaded ellipses and the associated cones depict sensor detection ranges. Each sensor uses HTTPS links to periodically retrieve server peppers from the central server and uses another HTTPS link to upload anonymized PRs. Time synchronization is achieved by calibration with NTP servers. Communication links are depicted for only one sensor to avoid clutter.

1.2. Collected Data and the Anonymization Procedure

As depicted in Figure 2, sensors extract three key data from each PR: (i) a timestamp (whose precision is of one second), (ii) a received signal strength indicator (RSSI) in dBm and (iii) a source address (SA) (MAC address). Although some smartphones randomize the SAs embedded in PRs, it is not guaranteed and we want user tracking to remain impossible, even without terminal-side SA randomization. Thus, we transform the original SA into an *SA identifier*, which is its anonymous counterpart.

To generate an *SA identifier* from an SA, we use a SHA-256 hash function in conjunction with a pepper and truncate its output to 64 bits. With $\{0,1\}^\gamma$ denoting the set of all binary sequences of γ bits, our anonymization function is $h : \mathcal{X} \to \{0,1\}^{64}$, which is a truncated SHA-256 hash function whose inputs are 48-bit SAs ($\mathcal{X} = \{0,1\}^{48}$). Note that generating SA identifiers of 64 bits is advantageous as such binary sequences can be easily stored as long integers in most databases (e.g., using the standard SQL `BIGINT` data type).

We prepend a time-varying pepper to every MAC address before hashing it. With $||$ denoting the concatenation operation, and `mac_address` and `global_pepper` representing, respectively, the MAC address (i.e., the SA) to be anonymized and the pepper prepended, $h(\texttt{global\_pepper}||\texttt{mac\_address})$ generates the SA identifier.
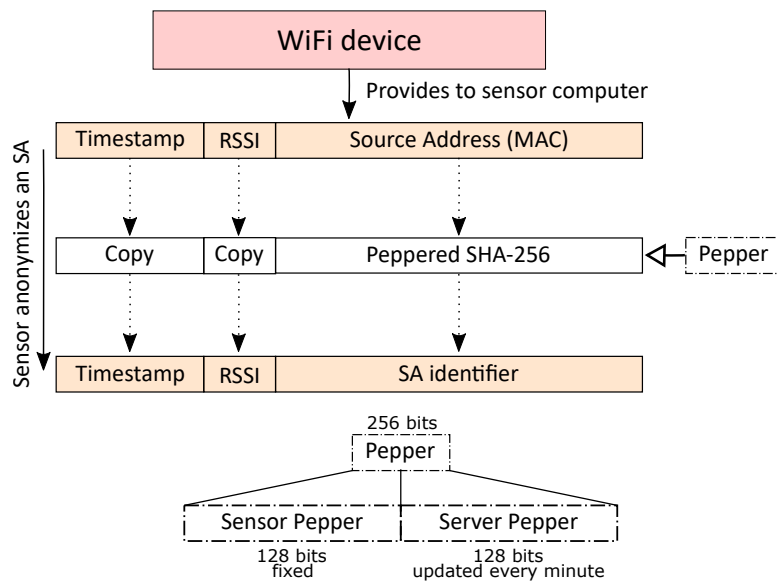
Figure 2. (From [5]) Scheme of the anonymization procedure executed by the sensors.

The pepper consists of a concatenation of a fixed 128-bit *sensor pepper* and a time-varying 128-bit *server pepper*. The central server maintains an up-to-date array of 20 server peppers for a duration of 20 min that sensors periodically fetch using an HTTPS link with transport security layer (TLS). Sensors use each server pepper for a specific *one-minute time frame*. Server peppers are generated using a pseudo random number generator (PRNG) (e.g., `/dev/urandom` or `/dev/random` on Linux). If this PRNG is deemed not secure (see [6]), hardware PRNG generators are alternatives [7,8].

The server and the sensors delete server peppers once they become outdated; in particular, the sensors erase the volatile memory chunk storing server peppers before updating it with new peppers periodically retrieved from the server.

The fixed sensor pepper forms a last line of defense in case the server peppers get compromised. It is written in a file or in the codebase of the sniffer, and it is never stored on the server. We have proposed a fixed sensor pepper but storing pregenerated sensor peppers for time frames of one minute is possible too; it would represent 42 MB of data for five years.

Loosely speaking, the time-varying and eventually forgotten pepper has a high entropy and breaking the anonymization scheme is about finding out its value for all one-minute time frames of interest. As we show thereafter, this procedure is not computationally tractable. We also explain why SA identifiers generated using different peppers cannot be compared against one another, thereby precluding user tracking. Moreover, despite the data distortion that our anonymization procedure entails, we also demonstrate that it does not affect in any significant way the output of our counting method (a procedure that we explained in more details in [3,5]). Intuitively, anonymization cannot affect counting if the SA identifier of any SA is identical across all sensors at (almost) all time instants.

It is also possible for $h : \mathcal{X} \to \{0,1\}^{64}$ to output (random) tokens, instead of being a truncated cryptographic hash function. In this case, the outputted tokens are truly uniformly distributed in the space $\{0,1\}^{64}$. The associated tokens should be kept in volatile memory (as well as the corresponding inputs) for a given anonymization window but can be wiped out once a new anonymization window begins. This random-token approach is typically well-suited to a central and final anonymization round. It would not be practical to carry it out on a distributed network because all nodes should then agree on a mapping from input SAs to tokens in real time.

1.3. Contributions

The following sections review the crowd monitoring system used in [3,4] for forecasting purposes and presented in [5] (with fewer details about anonymization than in this manuscript). This paper discusses the strength of our anonymization procedure and the effect of time synchronization inaccuracies on it. Besides the proposal of the anonymization process, our contributions also include the demonstration that our system satisfies the following four requirements:

1. It is computationally intractable to recover the original MAC addresses from the anonymous identifiers our system generates.
2. Anonymous identifiers from two distinct one-minute time frames cannot be compared against one another, which entails the impossibility to track individuals over time.
3. The proportion of time instants during which two sensors of our system could generate distinct anonymous identifiers for the same MAC address is negligible.
4. Assuming WiFi devices in an area generate 10^7 distinct MAC addresses within one minute in a monitored area, the collision rate of our anonymization procedure is lower than 10^{-9}. The value of 10^7 distinct MAC addresses corresponds roughly to an event of a few million people, which is comparable to or higher than the number of attendees of the vast majority of public events in the world.

Requirements (1) and (2) guarantee privacy, in that the original MAC addresses of devices cannot be recovered and also because tracking individuals is impossible. Requirements (3) and (4) enable the central server to compute accurate attendee counts. Should Requirement (3) not be met, sensors would too often return different SA identifiers for identical devices simultaneously detected (because of overlapping detection ranges), thereby inducing a positive counting bias. Requirement (4) ensures a negligible probability of two devices being identified as a single one (which would imply a negative counting bias).

Proving our system meets Requirement (4) is overwhelmingly a mathematical effort that is based on mathematical approximations of the collision rate of hash functions. This is the most complex result to derive in this paper and, due to its general nature, the theorem approximating the collision rate could be of interest to researchers pursuing other endeavors than the design of a crowd counting system.

1.4. Comparison with the State of the Art

The authors of ([9], Section 5) succinctly mentioned using random binary sequences appended to the MAC addresses prior to hashing (or to replace MAC addresses with tokens, more specifically, universally unique identifiers (UUIDs) [10]). Our anonymization scheme uses a similar idea, except that we prepend random sequences a central server partially generates and then shares with time-synchronized sensors. Each sequence is used simultaneously by all our sensors for one minute, a time after which the server and the sensors erase it. Thus, brute force attacks consist in recovering a pepper of high entropy instead of hashed MAC addresses, whose entropy is too low to withstand such attacks [9,11,12]. We also split peppers into two parts (which [9] does not propose), with one unknown to the server.

In [13], the authors developed a system similar to ours but for road traffic monitoring. Their anonymization scheme ([13], Section VI) relies on a truncation of the MAC address prior to hashing, whereas we rely on time-varying peppers of sufficiently high entropy to ensure anonymity and prevent brute-force attacks. Based on their experiments, it is unclear whether their anonymity scheme based on MAC address truncation would yield unacceptably high collision rates for large-scale crowds.

A very recent work [14] presented research similar to ours. In [14], the authors derived the collision rate we present in Theorem 1 ([14], Section 4.2) and also justified the interest of such a derivation within the framework of WiFi and Bluetooth signal detection for crowd counting. They also validated Theorem 1 numerically for a number of MAC addresses lower than 2×10^5 and for a number of output bits after hashing of up to 24 bits ([14], Section 1). Thanks to our precise approximation of the collision rate (see Theorem 2), we

can handle vastly higher values (e.g., a number of output bits of 64 bits and 10^7 MAC addresses). Moreover, our method is based on secret peppers that are forgotten and that are split in two parts: one stored on sensors and the other stored on a central server, so that if either the sensors or central server are compromised, anonymity still holds (see Section 2.1). The time-varying nature of our peppers also makes it impossible to track individuals (see Section 2.2). We also discussed the impact of typical time-synchronization errors on modern networks and found them to have no significant impact on the counting process we used in [3,4] (see Section 2.3). Finally, we point out that our (novel) approximation of the collision rate (and its analytical error bounds) are nontrivial mathematical results to derive (see Section 2.3 and the Appendix A).

Other related works on crowd counting using WiFi probe requests are [15,16]. In particular, Ref [15] discussed smartphone-executed MAC address randomization and its impact on crowd-counting algorithms. The authors also proposed a method for generating fingerprints that allowed them to track individuals whose smartphones emit PRs (a possibility that our system precludes on purpose for privacy reasons). The work [16] dealt with user positioning, especially in indoor environments and for nondense crowds. They notably improved positioning accuracy by leveraging signal strength indicators.

1.5. Outline

Section 1 has detailed the way our system works, with Section 1.4 comparing our results against the state of the art. Then, Section 2 shows that our four requirements are met. Finally, Section 3 is the conclusion. Appendix A contains mathematical proofs.

2. Results

We now turn to our contribution: proving our four requirements are met by the already existing crowd-counting system presented in [3–5]. We insist again that these results are new and not detailed in [3–5].

The data collection process is a means to an end: make it possible to count the number of people visiting an area while ensuring their privacy. In other words, the need for satisfying the four requirements is about ensuring two properties: privacy and accurate counting. The first two requirements address the former: how to ensure the privacy of users is preserved and tracking them (even anonymously) is impossible? The penultimate and last requirements deal with the second property: how to ensure that our privacy-enhancing data distortion does not affect counting accuracy? The next subsections detail our four requirements and show how our system satisfies them.

2.1. Requirement 1: Impossibility to Recover the Original SA from SA Identifiers

Cryptographic hash functions such as SHA-256 cannot be directly reversed; in practice, reversing consists of trying inputs until finding one whose hash is the output to be reversed. It is possible for an attacker to know the input MAC address of a particular entry in the list of anonymized PRs; for example, an attacker may go near sensors and send fake PRs with precise timing patterns that make it easy to identify them. In this case, brute forcing the pepper entails testing many of the 256-bit sequences that exist (on average, half of them should be tested). Attackers usually perform this operation using graphical processing units (GPUs), field-programmable gate arrays (FPGAs), or, if they have large resources, application-specific integrated circuits (ASICs). Let us examine if this attack is feasible with GPUs.

For example, 1 million Nvidia RTX 2080 SUPER Founders Edition graphics cards can compute roughly 5700 SHA-256 terahashes per second [17]; this implies that testing all 256-bit peppers (approximately 1.16×10^{65} terahashes) takes 2.04×10^{61} s, i.e., 6.47×10^{53} years. Should one of the two 128-bit peppers be known to an attacker, testing all 128-bit sequences still takes roughly 1.90×10^{15} years. We point out that relying on a regular SHA-256 hash function without peppers is not safe (see [9,12] and ([11], Section VI)) as the entropy of MAC addresses is too low to resist brute force attacks. We also highlight

that using computationally intensive hashes like *bcrypt* [18] and *Argon2* [19] would imply unreasonable computational requirements for sensors (see also ([9], Section 5)).

2.2. Requirement 2: Preventing Tracking for More Than One Minute

This requirement is linked to server peppers being updated between consecutive time frames of one minute. In particular, the avalanche effect of SHA-256 hash functions makes hashing with different peppers return incomparable SA identifiers for any fixed MAC address. (The avalanche effect of cryptographic hash functions is the fact that minor changes in the input significantly change the hash.)

2.3. Requirement 3: Peppers Are Identical Across All Sensors at a Given Time Instant

This requirement depends on the accuracy of time synchronization. We propose to use network time protocol (NTP), which implies accurate time synchronization on low-latency networks (e.g., 4G networks, with timing errors lower than 10 ms [20]). There could be synchronization-related mismatches at the frontiers of consecutive one-minute time frames but only for 20 ms/60,000 ms = 0.033% of their duration. Assuming probe request transmission times are uniformly distributed in time, this figure translates into having on average 0.033% of all PRs being anonymized by different peppers on the sensors.

2.4. Requirement 4: A Collision Rate of Less Than 10^{-9} for 10^7 MAC Addresses

We now derive estimates of the collision rate of truncated hash functions. The first part of this section is mathematical while the second part leverages the results of the first one to show the collision rate achieved by our system to be negligible for up to 10 million SAs.

2.4.1. Mathematical Foundations

Variable m denotes a number of possible outputs, such that $\log_2(m) \in \mathbb{N}$, and $\{0,1\}^\gamma$ denotes the set of all binary sequences of γ bits. We consider a function $h : \mathcal{X} \to \{0,1\}^{\log_2(m)}$ (with $n := \text{card}(\mathcal{X})$). Hereafter, h is a hash function, whose output is approximately uniformly distributed in $\{0,1\}^{\log_2(m)}$ ([21], Section 9.7.1). It could also be a token generator, in which case the uniform distribution assumption is exactly satisfied.

We follow the standard terminology in the study of hash tables and refer to m and n as the *number of buckets* and the *number of inserts*, respectively. Similarly, $\alpha := n/m$ is called the *load factor*. Finally, $Y^{(n,m)}$ denotes the (random) number of collisions when inserting n values into m buckets (with the uniform distribution assumption). Theorem 1 provides an exact—yet numerically unstable—formula of $\mathbb{E}\left[Y^{(n,m)}\right]$.

Theorem 1. *For n inserts into m buckets, the collision rate, $\mathbb{E}[Y^{(n,m)}]/n$, is*

$$\frac{\mathbb{E}\left[Y^{(n,m)}\right]}{n} = 1 - \frac{m}{n}\left(1 - \left(\frac{m-1}{m}\right)^n\right), \tag{1}$$

where the uniform distribution assumption has been used.

Proof. See the Appendix A. □

As shown in Figure 3, (1) suffers from numerical instabilities for sufficiently low values of the load factor. Therefore, for systems whose load factors are too low for (1) to provide accurate estimates, approximations are needed. In particular, to ensure such approximations are accurate enough, they should have proven analytical error bounds. Theorem 2 proposes three approximations of $\mathbb{E}[Y^{(n,m)}]/n$, with proven error bounds. Only the penultimate and last inequalities of Theorem 2 are numerically stable.
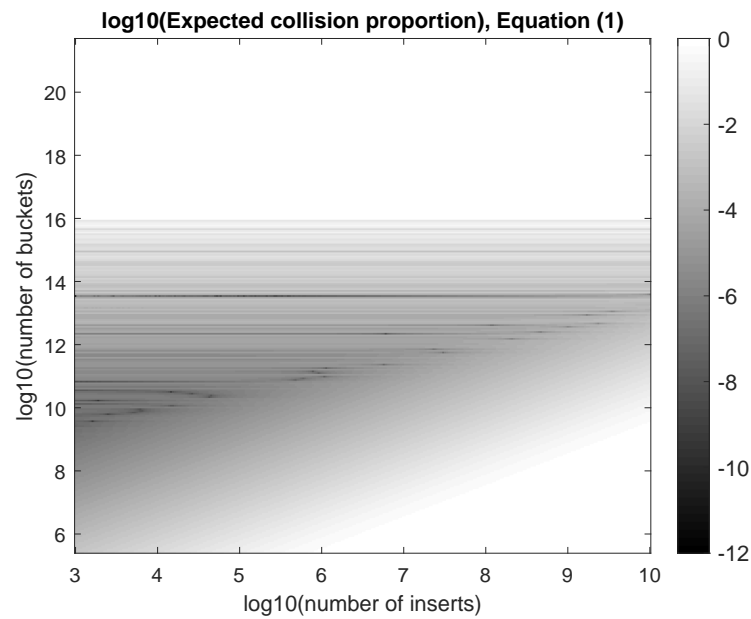
Figure 3. Numerically computed value of $\log_{10}(\mathbb{E}\left[Y^{(n,m)}\right]/n)$ (using (1)) in Matlab R2019a as a function of the number of inserts n and the number of buckets m. With $\log_{10}(n) \geq 3$, numerical instabilities appear for values of $\log_{10}(m)$ as low as 9.

Theorem 2. *For a degree of approximation $K \geq 2$, a number of inserts $n \geq 2$, and a load factor $\alpha \leq 1$, there exist error terms $\delta(\alpha, n)$ and $R_{K-1}(\alpha)$ such that*

$$\frac{\mathbb{E}\left[Y^{(n,m)}\right]}{n} = 1 - \alpha^{-1}(1 - \exp(-\alpha)) + \delta(\alpha, n) \tag{2}$$

$$= \sum_{k=1}^{K-1} \frac{\alpha^k (-1)^{k+1}}{(k+1)!} + \delta(\alpha, n) + R_{K-1}(\alpha) \tag{3}$$

$$= \frac{\alpha}{2} + \delta(\alpha, n) + R_1(\alpha), \tag{4}$$

where

$$-\sqrt{\frac{\alpha^2}{n^2 - \alpha^2}\left(\frac{\pi^2}{6} - 1\right)} \leq \delta(\alpha, n) \leq 0, \tag{5}$$

$$|R_{K-1}(\alpha)| \leq \frac{\alpha^K}{(K+1)!}, \tag{6}$$

and, in particular,

$$\frac{|R_1(\alpha)|}{\alpha/2} \leq \frac{\alpha}{3}. \tag{7}$$

Proof. See the Appendix A. □

2.4.2. The Interpretation of Theorem 2

Theorem 2 approximates the exact value of the collision rate that Theorem 1 provides. Equation (2) yields a first approximation that is not numerically stable for sufficiently low values of α (a figure similar to Figure 3 can be easily generated for (2) but has been omitted for the sake of brevity). Equation (3) provides a numerically stable approximation whose precision is controlled through K, hence the name "*degree of approximation*".

The error term $\delta(\alpha, n)$ quantifies to what extent $(1 - \alpha/n)^n$ accurately approximates $\exp(-\alpha)$. The term $R_{K-1}(\alpha)$ bounds the error tied to approximating $\exp(-\alpha)$ using its Kth-order Taylor polynomial, an approach used to derive (3) from (2).

For low values of α (e.g., $\alpha \leq 10^{-3}$), (4) is an accurate approximation because $|R_1(10^{-3})|/(10^{-3}/2) \leq 10^{-3}/3$ (see (7)), i.e., the error $|R_1(\alpha)|$ is less than 0.1% of the approximated value $\alpha/2$. For $\alpha \leq 1$ and for n high enough (say, $n \geq 100$), $\alpha^2/(n^2 - \alpha^2) \simeq \alpha^2/n^2 = 1/m^2$. Thus, with $m \geq 2^{64}$, $|\delta(\alpha, n)| \leq m^{-1}0.8031 \leq 5 \times 10^{-20}$.

2.4.3. Proving Requirement 4 Is Satisfied

We have $m = 2^{64} \simeq 1.84 \times 10^{19}$, which means that we truncate SHA-256 hashes to 64 bits. This corresponds to a load factor $\alpha = 10^7(1.84)^{-1}10^{-19} \simeq 10^{-12}$ for $n = 10^7$ MAC addresses. Figure 4 then shows that the collision rate expectation is approximately equal to $10^{-12.5}$. Note that, for α sufficiently low (e.g., $\alpha \leq 10^{-3}$), the approximation becomes (4), which explains why the level sets in Figure 4 appear to be linear slopes.

We point out that approximation errors are negligible for our choice of parameters. Our load factor $\alpha \simeq 10^{-12}$ implies (for any $K \geq 2$) $|R_{K-1}(\alpha)| \leq 10^{-24}$. Moreover, as already pointed out, $m \geq 2^{64} \implies |\delta(\alpha, n)| \leq 5 \times 10^{-20}$.

The conclusion is that our estimate of the collision rate expectation is approximately equal to $10^{-12.5}$, with an error upper bounded by $5 \times 10^{-20} + 10^{-24} \simeq 5 \times 10^{-20}$, so that Requirement 4 is met.
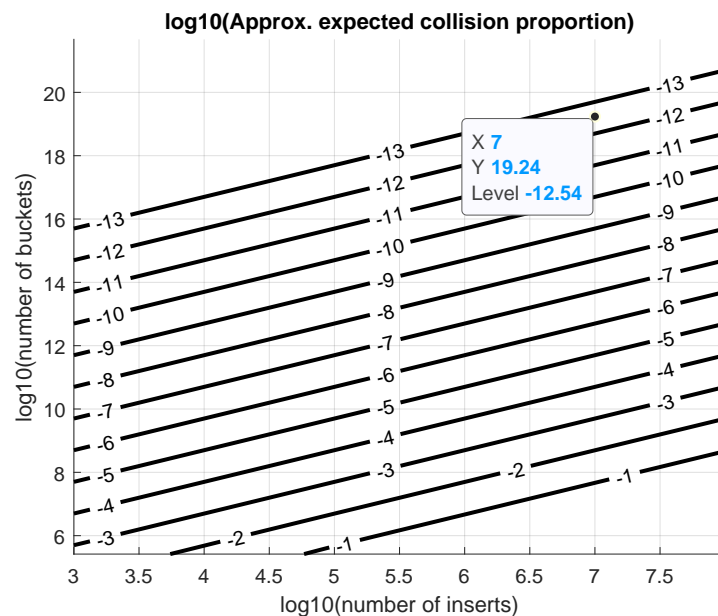

Figure 4. Level sets of the approximation (3) of the collision rate as a function of the number of inserts n and the number of buckets m.

2.4.4. Concentration Inequality for the Collision Rate

While it is interesting to find an upper bound for the expectation of the collision rate, $Y^{(n,m)}/n$, finding an upper bound for the probability that it exceeds some threshold is also a worthy endeavor. We propose such a (coarse) inequality. Because $Y^{(n,m)}/n \geq 0$, we can apply Markov's inequality:

$$\mathbb{P}\left[\frac{Y^{(n,m)}}{n} \geq a\right] \leq \frac{\mathbb{E}\left[Y^{(n,m)}/n\right]}{a}. \tag{8}$$

Using Theorem 2 with $K = 2$, we only know that $\mathbb{E}\left[Y^{(n,m)}/n\right] = \alpha/2 + \delta(\alpha, n) + R_1(\alpha)$ where $\delta(\alpha, n) \leq 0$ and $R_1(\alpha) \leq \alpha^2/6$. Therefore, we can only use the slightly more pessimistic concentration inequality that is

$$\mathbb{P}\left[\frac{Y^{(n,m)}}{n} \geq a\right] \leq \frac{\alpha/2 + \delta(\alpha, n) + R_1(\alpha)}{a} \leq \frac{\alpha/2 + \alpha^2/6}{a}, \tag{9}$$

where the term $\alpha^2/6$ is negligible in comparison to $\alpha/2$ for α sufficiently low (e.g., $\alpha \leq 10^{-3}$).

For example, let us consider again the previous calculation of Section 2.4.3 (with $n = 10^7$ MAC addresses, $m = 2^{64}$ and $\alpha = 10^{-12}$), which yielded $\mathbb{E}[Y^{(n,m)}/n] = \alpha/2 + \delta(\alpha, n) + R_1(\alpha) \simeq 10^{-12.5}$. Owing to $\alpha^2/6 \ll \alpha/2$ and with $a = 10^{-9}$,

$$\mathbb{P}\left[\frac{Y^{(n,m)}}{n} \geq 10^{-9}\right] \leq \frac{10^{-12.5}}{10^{-9}} = 10^{-3.5} \simeq 3.16 \times 10^{-4}, \tag{10}$$

which shows that, with probability 99.968%, the collision rate of our counting system does not exceed 10^{-9}.

Markov's inequality is coarse (and it may be possible to improve our result using a more sophisticated inequality) but, within the context of finding an upper bound for the collision rate of our crowd counting system for large crowds, that inequality is sufficient to prove its collision rate does not exceed 10^{-9} with high probability for large crowds (10^7 MAC addresses per minute).

2.5. Validating Requirement 4 Experimentally

An interesting future work endeavor would be to validate Requirement 4 experimentally and to evaluate how sharp the inequalities we obtained are. In particular, an interesting question is to determine to what extent the truncated SHA-256 hashes are close to being randomly distributed and how a discrepancy from uniformity translates into higher collision rates in our particular application. A conceptually simple analysis of this question could be carried out by generating a statistically significant number of random peppers and, for each pepper, to generate at least 10^{14} random SAs to evaluate the empirical collision rate (which we know should be around $10^{-12.54}$ according to Figure 4, which explains why generating at least 10^{14} SAs is statistically sound). Recent simulation results related to this approach are available in ([14], Section 5).

Unfortunately, rigorously validating the collision rate experimentally using datasets of true SAs would require monitoring events gathering millions of individuals. Moreover, it would be impossible to know exactly how many people carry smartphones and when each smartphones send PRs. As a result, we propose a slightly weaker variant (that still requires significant efforts). First of all, one needs to identify randomization and PR emission patterns from modern smartphones in a controlled laboratory environment (or use existing results on typical PR generation processes in the literature, see ([15], Figure 1)). This is equivalent to building a statistical distribution that accurately depicts the random process of modern smartphones generating PRs. Then, the methodology of the previous paragraph can be used with this distribution instead of a uniform one for SAs. The difficulties here mainly are about identifying PR transmission patterns for an extensive set of modern smartphones as well as evaluating what is the market share of each smartphone that is tested.

3. Conclusions

Within the framework of WiFi-based crowd counting, this paper proposed an anonymization scheme for collected MAC addresses. This anonymization scheme was endowed with four desirable properties. First, it made the recovery of original MAC addresses computationally intractable. Second, it precluded tracking capabilities. Third, it worked properly as long as timing synchronization errors between nodes collecting MAC

addresses were of the order of 10 ms, which is typically easy to attain on modern cellular networks. Fourth, it achieved a negligible collision rate between MAC addresses. This last point was supported by ample theoretical evidence. Although this paper was motivated by crowd counting applications, the methods and mathematical results could be of interest in other domains.

Author Contributions: Conceptualization, J.-F.D., S.A., F.H. and P.D.D.; methodology, J.-F.D., S.A., F.H. and P.D.D.; software, J.-F.D.; formal analysis, J.-F.D. and S.A.; writing—original draft preparation, J.-F.D.; writing—review and editing, J.-F.D., S.A., F.H. and P.D.D.; visualization, J.-F.D.; supervision, F.H. and P.D.D.; project administration, F.H. and P.D.D.; funding acquisition, J.-F.D., F.H. and P.D.D. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

ASIC	Application-specific integrated circuit
FPGA	Field-programmable gate array
GPU	Graphical processing unit
HTTPS	Hypertext tranfer protocol secure
MAC	Media access control
NTP	Network time protocol
PR	Probe request
PRNG	Pseudo random number generator
UUID	Universally unique identifier
RSSI	Received signal strength indicator
SA	Source address
TLS	Transport layer security

Appendix A. Proofs

In what follows, $\|x\|_2$ denotes the ℓ_2-norm of vector x. The notation $(a_k)_{1 \leq k \leq K}$ is equivalent to the vector $(a_1, a_2, \ldots, a_K)$ of size K.

Appendix A.1. Proof of Theorem 1

Let p_j denote the probability that the jth ($1 \leq j \leq m$) bucket be empty after n inserts. All inserts have equal probabilities to fall within each bucket and whether an insert ends up in one bucket is independent of which buckets are already occupied. As a result, we have $p_j = ((m-1)/m)^n$. Indeed, for the jth bucket to be unoccupied, all n inserts should end up in any of the other $m-1$ buckets and, for each insert, there is a probability $(m-1)/m$ that it ends up in any bucket except the jth one. The expectation of the number of empty buckets after n inserts is equal to

$$\sum_{j=1}^{m} \mathbb{E}[A_j] = \sum_{j=1}^{m} \left(\frac{m-1}{m} \right)^n = m \left(\frac{m-1}{m} \right)^n,$$

where $A_j = 1$ if the jth bucket is empty and equals 0 otherwise. Hence, the expectation of the number of occupied buckets is $m - m((m-1)/m)^n$. Without any collision after n inserts, there are exactly n distinct occupied buckets. However, with $n_l < n$ distinct occupied buckets, there are $n - n_l$ collisions. As the number of collisions is equal to $n -$ "number of occupied buckets" the average number of collisions is $n - m(1 - ((m-1)/m)^n)$ and the proof is complete.

Appendix A.2. Lemmas for Theorem 2

To prove Theorem 2, we first derive two lemmas. Lemma A1 quantifies to what extent $(1 - \alpha/n)^n$ is a good approximation of $\exp(-\alpha)$.

Lemma A1. *For $n \geq 1$ and $\alpha < n$,*

$$\left(1 - \frac{\alpha}{n}\right)^n = \exp(-\alpha)F(\alpha, n), \tag{A1}$$

where

$$\exp\left(-\alpha^2\sqrt{\frac{1}{n^2 - \alpha^2}\left(\frac{\pi^2}{6} - 1\right)}\right) \leq F(\alpha, n) \leq 1. \tag{A2}$$

Proof. For $0 \leq \alpha/n < 1$, using the Maclaurin series of $\log(1-x) = -\sum_{k=1}^{\infty} x^k/k$ (valid for $|x| < 1$), we obtain

$$\left(1 - \frac{\alpha}{n}\right)^n = \exp\left(n\log\left(1 - \frac{\alpha}{n}\right)\right)$$

$$= \exp\left(-n\sum_{k=1}^{\infty} \frac{(\alpha/n)^k}{k}\right)$$

$$= \exp\left(-\alpha\left(1 + \sum_{k=1}^{\infty} \frac{(\alpha/n)^k}{k+1}\right)\right), \tag{A3}$$

where we have used

$$-n\sum_{k=1}^{\infty} \frac{(\alpha/n)^k}{k} = -\sum_{k=1}^{\infty} \frac{\alpha^k}{n^{k-1}k} = -\alpha\left(1 + \sum_{k=2}^{\infty} \frac{\alpha^{k-1}}{n^{k-1}k}\right) = -\alpha\left(1 + \sum_{k=1}^{\infty} \frac{\alpha^k}{n^k(k+1)}\right).$$

Defining $f^{(K)}(\alpha, n) := \sum_{k=1}^{K}(\alpha/n)^k/(k+1)$, we have, $0 < f^{(1)}(\alpha, n) < f^{(2)}(\alpha, n) < \cdots$ so that if for all K, $f^{(K)}(\alpha, n) \leq \xi(\alpha, n)$, then $\sum_{k=1}^{\infty}(\alpha/n)^k/(k+1) \leq \xi(\alpha, n)$. The sum in $f^{(K)}(\alpha, n)$ is the inner product between vectors $((\alpha/n)^k)_{1 \leq k \leq K}$ and $(1/(k+1))_{1 \leq k \leq K}$. The Cauchy–Schwarz inequality yields:

$$f^{(K)}(\alpha, n) \leq \sqrt{\left\|\left(\frac{\alpha^k}{n^k}\right)_{1 \leq k \leq K}\right\|_2^2 \left\|\left(\frac{1}{k+1}\right)_{1 \leq k \leq K}\right\|_2^2}.$$

We have, using an asymptotic expression for geometric series,

$$
\left\| \left(\frac{\alpha^k}{n^k} \right)_{1 \leq k \leq K} \right\|_2^2 = \sum_{k=1}^{K} \left(\left(\frac{\alpha}{n} \right)^k \right)^2
$$

$$
= \sum_{k=0}^{K} \left(\left(\frac{\alpha}{n} \right)^2 \right)^k - 1
$$

$$
\leq \sum_{k=0}^{\infty} \left(\left(\frac{\alpha}{n} \right)^2 \right)^k - 1
$$

$$
= \frac{1}{1 - \alpha^2/n^2} - 1
$$

$$
= \frac{\alpha^2}{n^2 - \alpha^2}.
$$

Moreover,

$$
\left\| \left(\frac{1}{k+1} \right)_{1 \leq k \leq K} \right\|_2^2 = \sum_{k=1}^{K+1} \frac{1}{k^2} - 1
$$

$$
\leq \sum_{k=1}^{\infty} \frac{1}{k^2} - 1
$$

$$
= \zeta(2) - 1,
$$

where $\zeta(2)$ is the Riemann zeta function evaluated at 2, which is equal to $\pi^2/6$. Therefore, we may use the upper bound

$$
\xi(\alpha, n) := \alpha \sqrt{\frac{1}{n^2 - \alpha^2}} \sqrt{\frac{\pi^2}{6} - 1}. \tag{A4}
$$

It is also easy to notice that $\sum_{k=1}^{\infty} (\alpha/n)^k/(k+1) \geq 0$ given that all the terms of the sum are positive.

Injecting these results in (A3), we obtain

$$
\left(1 - \frac{\alpha}{n} \right)^n = \exp\left(-\alpha \left(1 + \lim_{K \to \infty} f^{(K)}(\alpha, n) \right) \right)
$$

$$
= \exp(-\alpha) \exp\left(-\alpha \lim_{K \to \infty} f^{(K)}(\alpha, n) \right)
$$

$$
= \exp(-\alpha) F(\alpha, n)
$$

where

$$
F(\alpha, n) \leq \exp(0) = 1
$$

and

$$
F(\alpha, n) \geq \exp\left(-\frac{\alpha^2}{\sqrt{n^2 - \alpha^2}} \sqrt{\frac{\pi^2}{6} - 1} \right)
$$

because $\lim_{K \to \infty} f^{(K)}(\alpha, n) \leq \xi(\alpha, n)$ according to (A4). □

We now turn to a lemma focusing on the accuracy of a polynomial approximation of $\alpha^{-1}(1 - \exp(-\alpha))$.

Lemma A2. *For $0 < \alpha \leq 1$, $K \geq 1$ and $g : [0,1] \to \mathbb{R} : \alpha \mapsto g(\alpha) = \alpha^{-1}(1 - \exp(-\alpha))$,*

$$g(\alpha) = \sum_{k=0}^{K-1} \frac{\alpha^k}{(k+1)!} (-1)^k + R_{K-1}(\alpha)$$

where

$$|R_{K-1}(\alpha)| \leq \frac{\alpha^K}{(K+1)!}$$

Proof. With $\ell(\alpha) := -\exp(-\alpha)$, it is easy to compute that

$$\frac{\mathrm{d}^k \ell}{\mathrm{d}\alpha^k}(x) = (-1)^{k+1} \exp(-x).$$

Thus,

$$\max_{x \in [0,1]} \left| \frac{\mathrm{d}^{K+1}\ell}{\mathrm{d}\alpha^{K+1}}(x) \right| = 1. \tag{A5}$$

Taylor's theorem ([22], Theorem 5.15) shows that the Kth-order Taylor polynomial of $\ell(\alpha)$ around zero has a remainder $R'_K(\alpha)$, for which $|R'_K(\alpha)| \leq \alpha^{K+1}/(K+1)!$ over $\alpha \in [0,1]$ because of (A5). The desired $(K-1)$th-order polynomial approximation is:

$$\alpha^{-1}(1 - \exp(-\alpha)) = \alpha^{-1}\left(1 - \sum_{k=0}^{K} \frac{\alpha^k}{k!}(-1)^k - R'_K(\alpha) \right)$$

$$= \sum_{k=0}^{K-1} \frac{\alpha^k}{(k+1)!}(-1)^k + R_{K-1}(\alpha),$$

and the $(K-1)$th-order remainder is $R_{K-1}(\alpha) := -\alpha^{-1}R'_K(\alpha)$ and satisfies $|R_{K-1}(\alpha)| \leq \alpha^K/(K+1)!$. □

Appendix A.3. Proof of Theorem 2

Using Theorem 1, $\alpha = n/m$, $1/m = \alpha/n$ and Lemma A1, we derive

$$\frac{\mathbb{E}\left[Y^{(n,m)}\right]}{n} = 1 - \frac{m}{n}\left(1 - \left(\frac{m-1}{m} \right)^n \right)$$

$$= 1 - \alpha^{-1}\left(1 - (1 - \alpha/n)^n \right)$$

$$= 1 - \alpha^{-1}(1 - \exp(-\alpha)F(\alpha, n)). \tag{A6}$$

For $n \geq 2$ and $\alpha < 1$, $\mu(\alpha, n) := \alpha^2 \sqrt{\dfrac{1}{n^2 - \alpha^2}\left(\dfrac{\pi^2}{6} - 1 \right)}$ is monotonically decreasing with n and monotonically increasing with α, and it is approximately equal to $0.4637 < 1$ for $n = 2$ and $\alpha = 1$. We use the inequality $1 - x \leq \exp(-x)$ (valid for $x < 1$), with $x := \mu(\alpha, n)$, thereby implying $1 - \mu(\alpha, n) \leq \exp(-\mu(\alpha, n))$ because $\mu(\alpha, n) < 1$ for $n \geq 2$. Thus, from (A2) of Lemma A1, we derive

$$1 - \alpha^2 \sqrt{\frac{1}{n^2 - \alpha^2}\left(\frac{\pi^2}{6} - 1 \right)} \leq F(\alpha, n) \leq 1. \tag{A7}$$

Therefore, by combining (A6) and (A7), we obtain

$$\frac{\mathbb{E}\left[Y^{(n,m)}\right]}{n} \leq 1 - \alpha^{-1}\left(1 - \exp(-\alpha)F(\alpha, n) \right)|_{F(\alpha,n)=1} = 1 - \alpha^{-1}(1 - \exp(-\alpha)) \tag{A8}$$

and

$$\frac{\mathbb{E}\left[Y^{(n,m)}\right]}{n} \geq 1 - \alpha^{-1}(1 - \exp(-\alpha)F(\alpha, n)) \text{ with } F(\alpha, n) = 1 - \alpha^2 \sqrt{\frac{1}{n^2 - \alpha^2}\left(\frac{\pi^2}{6} - 1\right)}$$

$$= 1 - \alpha^{-1}(1 - \exp(-\alpha)) - \alpha^{-1}\alpha^2 \exp(-\alpha)\frac{1}{\sqrt{n^2 - \alpha^2}}\sqrt{\frac{\pi^2}{6} - 1}$$

$$= 1 - \alpha^{-1}(1 - \exp(-\alpha)) - \exp(-\alpha)\sqrt{\frac{\alpha^2}{n^2 - \alpha^2}}\sqrt{\frac{\pi^2}{6} - 1}$$

$$\geq 1 - \alpha^{-1}(1 - \exp(-\alpha)) - \sqrt{\frac{\alpha^2}{n^2 - \alpha^2}}\sqrt{\frac{\pi^2}{6} - 1}, \tag{A9}$$

where the last line stems from $-\exp(-\alpha) \geq -\exp(0) = -1$ for $\alpha \in [0, 1]$. As a result, combining (A8) and (A9), we get

$$-\sqrt{\frac{\alpha^2}{n^2 - \alpha^2}}\sqrt{\frac{\pi^2}{6} - 1} \leq \frac{\mathbb{E}\left[Y^{(n,m)}\right]}{n} - \left(1 - \alpha^{-1}(1 - \exp(-\alpha))\right) \leq 0, \tag{A10}$$

which provides the bounds of the theorem (Equation (2)) for the error term $\delta(\alpha, n)$. Then, Lemma A2 implies

$$1 - \alpha^{-1}(1 - \exp(-\alpha)) = 1 - \sum_{k=0}^{K-1}\frac{\alpha^k}{(k+1)!}(-1)^k - R_{K-1}(\alpha)$$

$$= \sum_{k=1}^{K-1}\frac{\alpha^k}{(k+1)!}(-1)^{k+1} - R_{K-1}(\alpha).$$

Injecting this last result into (2) proves (3). Deriving (4) and (7) is straightforward.

References

1. Martella, C.; Li, J.; Conrado, C.; Vermeeren, A. On current crowd management practices and the need for increased situation awareness, prediction, and intervention. *Saf. Sci.* **2017**, *91*, 381–393. [CrossRef]
2. Uras, M.; Cossu, R.; Ferrara, E.; Liotta, A.; Atzori, L. PmA: A real-world system for people mobility monitoring and analysis based on Wi-Fi probes. *J. Clean. Prod.* **2020**, *270*, 122084. [CrossRef]
3. Determe, J.F.; Singh, U.; Horlin, F.; De Doncker, P. Forecasting Crowd Counts With Wi-Fi Systems: Univariate, Non-Seasonal Models. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 6407–6419. [CrossRef]
4. Singh, U.; Determe, J.F.; Horlin, F.; De Doncker, P. Crowd Forecasting based on WiFi Sensors and LSTM Neural Networks. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 6121–6131. [CrossRef]
5. Determe, J.F.; Azzagnuni, S.; Singh, U.; Horlin, F.; De Doncker, P. Monitoring Large Crowds With WiFi: A Privacy-Preserving Approach. *IEEE Syst. J.* **2022**, 1–12 . [CrossRef]
6. Dodis, Y.; Pointcheval, D.; Ruhault, S.; Vergniaud, D.; Wichs, D. Security analysis of pseudo-random number generators with input: /dev/random is not robust. In Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, Berlin, Germany, 4–8 November 2013; pp. 647–658.
7. Stipčević, M.; Rogina, B.M. Quantum random number generator based on photonic emission in semiconductors. *Rev. Sci. Instruments* **2007**, *78*, 045104. [CrossRef]
8. Zheng, Z.; Zhang, Y.; Huang, W.; Yu, S.; Guo, H. 6 Gbps real-time optical quantum random number generator based on vacuum fluctuation. *Rev. Sci. Instruments* **2019**, *90*, 043105. [CrossRef] [PubMed]
9. Demir, L.; Cunche, M.; Lauradoux, C. Analysing the privacy policies of Wi-Fi trackers. In Proceedings of the 2014 Workshop on Physical Analytics, Bretton Woods, NH, USA, 16 June 2014 ; pp. 39–44.
10. Leach, P.; Mealling, M.; Salz, R. A Universally Unique Identifier (UUID) URN Namespace. 2005. Available online: https://www.rfc-editor.org/rfc/pdfrfc/rfc4122.txt.pdf (accessed on 18 April 2022).
11. Demir, L.; Kumar, A.; Cunche, M.; Lauradoux, C. The pitfalls of hashing for privacy. *IEEE Commun. Surv. Tutorials* **2017**, *20*, 551–565. [CrossRef]
12. Marx, M.; Zimmer, E.; Mueller, T.; Blochberger, M.; Federrath, H. Hashing of personally identifiable information is not sufficient. *SICHERHEIT 2018* **2018** . [CrossRef]

13. Fuxjaeger, P.; Ruehrup, S.; Paulin, T.; Rainer, B. Towards privacy-preserving Wi-Fi monitoring for road traffic analysis. *IEEE Intell. Transp. Syst. Mag.* **2016**, *8*, 63–74. [CrossRef]

14. Ali, J.; Dyo, V. Practical Hash-based Anonymity for MAC Addresses. In Proceedings of the 17th International Joint Conference on e-Business and Telecommunications, ICETE 2020—Volume 2: SECRYPT, Lieusaint, Paris, France, 8–10 July 2020; Samarati, P., di Vimercati, S.D.C., Obaidat, M.S., Ben-Othman, J., Eds.; ScitePress: Setúbal, Portugal, 2020; pp. 572–579. [CrossRef]

15. Hong, H.; De Silva, G.D.; Chan, M.C. Crowdprobe: Non-invasive crowd monitoring with Wi-Fi probe. *Proc. ACM Interactive Mobile Wearable Ubiquitous Technol.* **2018**, *2*, 1–23. [CrossRef]

16. Potortì, F.; Crivello, A.; Girolami, M.; Barsocchi, P.; Traficante, E. Localising crowds through Wi-Fi probes. *Ad Hoc Netw.* **2018**, *75*, 87–97. [CrossRef]

17. Nvidia RTX 2080 SUPER FE Hashcat Benchmarks. Available online: https://gist.github.com/epixoip/47098d25f171ec1808b519615be1b90d (accessed on 13 August 2020).

18. Provos, N.; Mazieres, D. A Future-Adaptable Password Scheme. In Proceedings of the USENIX Annual Technical Conference, FREENIX Track, Monterey, CA, USA, 6–11 June 1999; pp. 81–91.

19. Biryukov, A.; Dinu, D.; Khovratovich, D. Argon2: New generation of memory-hard functions for password hashing and other applications. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbruecken, Germany, 21–24 March 2016; pp. 292–302.

20. Miškinis, R.; Jokubauskis, D.; Smirnov, D.; Urba, E.; Malyško, B.; Dzindzelėta, B.; Svirskas, K. Timing over a 4G (LTE) mobile network. In Proceedings of the 2014 European Frequency and Time Forum (EFTF), Neuchatel, Switzerland, 23–26 June 2014; pp. 491–493.

21. Menezes, A.J.; Katz, J.; Van Oorschot, P.C.; Vanstone, S.A. *Handbook of Applied Cryptography*; CRC Press: Boca Raton, FL, USA, 1996.

22. Rudin, W. *Principles of Mathematical Analysis*; McGraw-hill: New York, NY, USA, 1964; Volume 3.