DOI: 10.15439/2022F123
ISSN 2300-5963 ACSIS, Vol. 30

Anomaly detection on compressed data in resource-constrained smart water meters

Sarah Klein
Sirris, Brussels, Belgium
Email: sarah.klein@sirris.be

Anna Hristoskova
Sirris, Brussels, Belgium

Annanda Rath
Sirris, Brussels, Belgium

Renaud Gonce
Shayp, Brussels, Belgium

***Abstract*—Increasing amount of devices in our daily life are equipped with sensors that transfer information to a cloud solution where the data is finally analysed. By improving the data intelligence on the edge, the data transfer can be reduced, which not only saves bandwidth and thus reduces energy consumption, but also leads to increased privacy protection. In this paper, we propose a privacy-friendly water leakage detection approach for various kind of water meters (optical and digital) performed on a very constrained, wireless devices.**

I. Introduction

THE amount of IoT devices on our homes increases for several years already. While most of them promise easier living and higher comfort, the actual applications are often neither environment nor privacy friendly. In this paper, we introduce a water leakage detection algorithm that takes both into account by running analytics directly on a long-living, energy-saving device in a privacy-preserving manner. For this, we apply a very lightweight data compression at the edge that enables leakage detection and significantly reduces the bandwidth needed to transfer the data from the edge to a central cloud device. By this, public as well as private buildings all over the world can profit from a solution that is easy to install and that reduces the waste of fresh water. This work was performed in close collaboration between Sirris and Shayp, who are leading specialists in water efficiency and monitoring, such that the evaluation of the approach was performed on real-world data.

The remainder of this paper is organized as follows: We will first provide an overview of current solutions in section II, before describing the technical setup in section III. In section IV we will explain in detail our solution for an on-the-edge leakage detection that respects privacy aspects. Its evaluation will be discussed in V-1 on both, artificial and real-world data before concluding in section VI.

II. Related work

Recently, several solutions on leakage detection by smart meters were suggested. The common approach entails a connected device at water meter level, which sends pulse data to a cloud back-end for further collection and analysis. Hence, the leakage detection is executed only in the cloud [1], [2], [3]. In [1], a solution for remote monitoring of water consumption and leakage detection is discussed. The setup consists of a water meter is connected to an Arduino micro-controller that sends the data to a Raspberry Pi gateway. Only from there, the data is hourly transferred to a cloud back-end that the end user can consult. From a hardware point of view, this solution is error prone as the two-fold messaging can lead to data loss and increased latency. Further, a wall socket has to be available in the vicinity of the water meter which makes it hard to generalise the solution.

The leakage detection suggested by the authors is performed at cloud level and consists of the detection of four different scenarios: (i) a negative consumption trend, which indicates a problem with the data transfer; (ii) a continuous water flow over 24 hours, which indicates a strong leakage; (iii) a coincidence with the last two measurements, which also indicates a leakage; and finally (iv) a significant deviation from the historical consumption. However, this solution suffers from a cold-start problem for detecting leakages reliably. Further, some of the rules are only meaningful in residential buildings, as e.g. hospitals can indeed show a significant hourly water consumption throughout an entire day.

Similarly, in [3], a wireless open source middleware was developed. Also here, the data is collected locally via an edge gateway but the leakage detection is only performed at cloud level. The empiric algorithm looks for the absence of water consumption during a specific interval across the day that deviates from historical consumption. With this rather rough estimate, it is possible to detect leakages in residential buildings, but it will probably fail in other types of buildings like hospitals, which were not investigated in the publication.

The solution presented in [4] makes use of wireless, battery-driven vibration sensors, instead of directly measuring the water consumption. Increasing vibrations in a pump indicate a pipe burst in the network. The proposed system uses a lightweight edge anomaly detection algorithm based on compression rates. The leakage detection splits the data x from the stream into two equally long sequences of length w_{stream}, e.g. at time t, the data sequences consist of $[x(t-2w_{stream},...,x(t-w_{stream})]$ and $[x(t-w_{stream},...,x(w_{stream})]$. For both, they apply miniLZO compression [5]. When the compression rate changes significantly from one window to the next, they assume that a leakage occurred. By using lossless compression in their lab-based test rig, the authors could reduce communication by 90% compared to periodical messaging which leads to an increasing battery lifetime. While this algorithm offers very suitable performance for big pipes and big bursts, it does not detect small leakages that are most common in residential households or schools [6]. Further, the authors do not consider privacy or security aspects critical when reducing the amount of messages sent. In case of a residential building, the number of messages being sent can already leak private information on the consumption pattern of this particular household. Lack of messages poses a more critical security thread as one can assume people are away.

III. Technical setup

Shayp's solution for monitoring water consumption and detecting anomalies in water usage consists of a wireless water meter reading device. It has long-range, battery-powered data logger compatible with all pulse-ready meters, specifically designed to withstand water immersion, harsh conditions and ensure an ideal performance in deep indoor situations. This is enabled using a Narrowband IoT (NB-IoT) connection to a back-end cloud. In addition of being low-power, this radio standard also offers a very good connectivity, allowing communication inside deep basements where water meters are usually to be found. Coupled to the periodic sending schema as explained below and thanks to the low-power micro-controller of the device, NB-IoT allows to meet a 10-year battery lifetime.

Through a pulse emitter placed on the water meter, the water consumption takes the form of pulses, each of which, depending on the water meter, corresponds to a certain amount of water (typically 1L or 10L) consumed per defined time window. A sensor connected to the pulse emitter detects these pulses. The data logger collects these pulses and sends this consumption data to a cloud back-end where water consumption analysis and leak detection take place.

1) Periodic sending schema: Shayp's device aggregates hourly consumption data in periods of 30 s and then sends a message to the cloud. Although the message length is 512 bytes, the actual used space is about 152 bytes (32 bytes for payload and 120 bytes for 120 data points). This results in 360 unused bytes in a message.

2) Cloud solution for leakage detection: Currently, Shayp supports leakage detection at the cloud level. For residential buildings this takes between 1-3 hours, while corporate buildings vary from 3 up to 24 hours. This is due to the complex water consumption pattern for buildings such as schools and hospitals, where water usage is constant. The lack of on-device analytics and the hourly message pattern prevent any anomaly detection in less than an hour. The data made available for this research consists of the number of consumption pulses and the estimated leakage size in pulses per 30 seconds. Further, information about the type of building, e.g. a public building or a private household, is provided.

With the method described in this publication, we improve the detection time even further (<1h for residential, <3h for corporate) by applying a lightweight analysis executed on the device at near real-time. This should however not hamper the lifetime of the battery, which ideally could even be extended to up to 16 years, which is the expected life time of water meters.

IV. Leakage detection

In this work, we introduce a leakage detection algorithm operating on a very constrained edge device based on data compression. The algorithm has to fulfill the following requirements in order to be applicable in real-world scenarios:

1) Minimal power consumption in order to guarantee at least 16 years of battery lifetime.
2) Fast leakage detection such that a warning can be sent with short delay (<1h for residential, <3h for corporate buildings).
3) Preserve the privacy of the underlying data

Interestingly, the three points above contradict each other: i.e. in order to increase the battery lifetime, the device should send as few messages as possible. But with less messages, it is harder to detect leakages early enough. One solution could be to only send messages once water consumption was measured and a risk of leakage was detected. Though, under this assumption, privacy and security in households or corporate buildings are at risk, as in this case an attacker can easily derive consumption patterns by profiling the message sending patterns.

In order to deal with these contradicting requirements, we developed a leakage risk assessment at the edge based on the compression of the consumption data. This approach is combined with a leakage-sensitive random messaging schema in order to ensure privacy preservation.

1) Data Consumption Compression: Per message, 480 bytes of data can be sent. In order to use this space as efficiently as possible, while not losing information, we apply a lightweight lossless sequential compression. We either use:

1) Fibonacci codes [7] on the unprocessed sequence, as it has proven to be very robust and efficient [8].
2) A combination of run-length encoding (RLE) [9] and Fibonacci codes where in a first step we apply RLE and only after Fibonacci codes.

For each device, more than 200.000 data points were used. One can see that the overall performance of the combined compression (RLE + Fibonacci) leads to better results, hence higher compression rates, for most devices (Fig. IV-1). Nevertheless, two extreme outliers can easily be detected, i.e. the two devices at the bottom rows, namely *NE83A580* and *N389F2E8*. By analysing the statistics of the consumption and leakage pulses given in Table I, it is possible to explain why the compression is so different in these cases: For (*N389F2E8*), the overall consumption is high such that the raw Fibonacci encoding is less effective as the strings encoding the integers are becoming longer. The mean leakage pulses for this device is NaN, as no leakage was detected by the leakage detection algorithm that is used in production. For *NE83A580*, the difference in compression between the sole Fibonacci encoding and combined compression indicates that the consumption values are fluctuating a lot such that the hardly any longer periods of

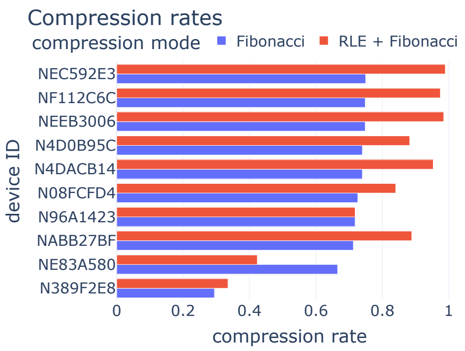

Fig. 1. Compression ratio for all devices for the two different compression approaches.

TABLE I
OVERALL STATISTICS OF CONSUMPTION AND LEAKAGE PULSES

device ID	consumption pulses	leak pulses
NEC592E3	0.004244	0.000005
NF112C6C	0.015171	0.004336
NEEB3006	0.016815	0.000212
N4D0B95C	0.085269	0.017191
N4DACB14	0.113889	0.000000
N08FCFD4	0.226471	0.045114
N96A1423	0.308077	0.113322
NABB27BF	0.475513	0.027130
NE83A580	0.784921	NaN
N389F2E8	21.086333	0.210611

the same (or no) consumption are given in the data. This gives a hint for (continuous) leakage in the data as also indicated by the high number of mean leakage pulses given in Table I. We will in more detail discuss the results per device in subsection V-2.

2) Anomaly detection: In most production-ready solutions, leakages in water supply are detected at cloud level [3], [1] where sufficient computational resources and historical consumption data are available. The approach we present here aims at estimating the risk of a leakage at edge level as early as possible. For this, we use the specific consumption patterns of a leaking asset. In case of a leakage, the consumption pulses $c(t)$ of a specific strength occur very regularly - the consumption of a leakage. Mathematically speaking, we define $\delta = |F(c(t_m))||RLF(c(t_m))|^{-1}$, where $|F(c(t_m))|$ is the length of the Fibonacci code of the consumption sequence $c(t_m)$ during a time window t_m, and $|RLF(c(t_m))|$ is the length of the combined compression (RLE + Fibonacci encoding) of the same sequence. Based on δ, we define the binary variable $L(c(t_m))$ indicating whether a leakage was detected in sequence $c(t_m)$ as being 1 for $\delta > \epsilon$ and 0 otherwise. where ϵ, the leakage threshold, is a free parameter of the model. This inflation given by a high δ-value can be used for leakage detection. In the case of a comparably strong leakage, the combination of RLE and Fibonacci codes is about twice as long as the sequences encoded by Fibonacci codes.

The overall idea is similar to the one in [4] but while they compare the compression in two consecutive windows, we compare the compression of the data within the same window with two different compression algorithms. This does not only reduce the possible alarm delay as our window can be shorter, it also takes into account the specific consumption pattern in case of a leakage.

3) Random sending schema: The sending schema is crucial in order to fulfill the requirements listed in the beginning of Section IV. When sending as few messages as possible the battery lifetime is strongly extended. The simplest solution is thus sending a message once the message space of 480 bytes is filled or a specific anomaly is detected. However, water consumption in residential buildings, similar to electricity consumption [10], is highly privacy sensitive. Even if we assume that the messages are properly and strongly encrypted, an attacker can deduce information on the consumption pattern and the presence of inhabitants by only counting the number of messages per time interval. Though a schema in which messages are sent in regular intervals prevents this possible privacy breach, it should be avoided: In order to increase the battery lifetime, the regular sending interval should be as large as possible but this leads to a significantly delayed leakage detection. For this reason, we suggest a privacy-preserving sending schema based on random timing as sketched, described in detail below. The algorithm takes the following variables as input:

- T: the expectation time for sending a message,
- ϵ: The leakage threshold as defined above
- ω: the look-back window on which to calculate the leakage risk, e.g. 2 hours
- $n_{\max}$: the maximal warnings to send per detected leakage
- R: the *force* radius
- ζ: a constant that defines noise in the sending pattern.
- b: The maximal sending time, e.g. 24 hours.

In a first step, we draw a random sending time interval t_i for $i \in \mathbb{N}_0$ from a truncated exponential distribution $f_T(t|\lambda, b)$ for $0 < t \leq b$ where $\lambda = \frac{1}{T} > 0$ and save it to the list of *random times*. We use a truncated function in order to ensure that messages are sent within time b (e.g. 24h). Every time a new sensor measurement is available (in our case every 30s), we run the following steps:

1) Calculate the encoding values of the sequence in the look-back window $F(c(\tau^i_j))$ and $RLF(c(\tau^i_j))$, where τ^i_j is the time of the j-th measurement in the i-th message.
2) If $\tau^i_j = t_i$, hence the time when the message should be sent, send the message with content (leakage risk, encoded sequence) and then calculate the average of the last R sending times from the *random times* list. We use this average sending time $T^\star$ in order to calculate the next random sending time t_{i+1} drawn from the distribution

$$f(t_{i+1}|\lambda, b) = \frac{\frac{1}{\lambda}\exp\left(\frac{1}{\lambda}\right)}{1 - \exp\left(-\frac{b}{\lambda}\right)} \text{ with } \lambda = T + K\frac{T - T^\star}{T^\star}. \quad (1)$$

We can see this similarly to a canonical description of a confined ideal gas in an external potential [11], where

T is the equilibrium value and the potential is given by the earlier sending of a message due to a detected leakage. The system is pushed back to equilibrium with the artificial temperature K. We then add the calculated next sending time t_{i+1} to the list of random sending times and set $i = i + 1$ and $j = 0$.

3) Otherwise, if $\tau_j^i \neq t_i$, calculate δ from $F(c(\tau_j^i))$ and $RLF(c(\tau_j^i))$. In case $\delta > \epsilon$, hence $L(c(t_m)) = 1$ and the number of sent warnings $n < n_{\max}$, draw a normally distributed random number $r = \mathcal{N}(\zeta, 0.2\zeta)$. In case $\tau_j^i < (n+1)r$, send the message. We use this in order to add an additional level of randomness to the sending pattern as well as to prevent too many warnings for the same detected and continuous leakage. The factor $n + 1$ gives here an additional damping. Just as above, calculate the new random sending time as given in eq. (1) and add it to the list of sending times.
4) In all other cases, wait for the next measurement.

In the next section, we will evaluate our approach on artificial as well as real-world data. We will judge our method by the accuracy of the detected leakages, the number of sent messages, which indicates the battery consumption on the device and will further perform an analysis on the distribution of sending times.

V. Evaluation

In this section, we will evaluate the proposed leakage detection and messaging algorithm on the edge with respect to the leakage detection accuracy and privacy preservation. In order to perform the analysis of the leakage detection performance, we will not only use the real-world consumption data but also artificially created data as in that case the actual starting point of the leakage is exactly known and we can perform exact statistics on leakage detection time. For privacy evaluation, we will use the real-world data only.

1) Artificial data: In order to create the artificial data, we extract the daily consumption per weekday from the data without leakages. For this, we first manually remove the consumption pulses related to leakages from the time series data from device *N96A1423*, which is a school building. Hence, the general consumption patterns for weekdays are very different from those on weekends but also the overall consumption per day is quite irregular. For each day of one year of artificial data, we select every 30 seconds randomly a consumption value from the same time and day of week from the historical consumption. We add some normally distributed noise from $r = \mathcal{N}(0, 0.2)$ and round the resulting value to an integer. In this way, the overall consumption pattern per day of the week is kept. Based on this consumption without leakages, we add pulses of leakages of different severity (S), different strength (s) and different length (Δt) to create the final artificial consumption. The severity S defines how strong the leakage is, i.e. if $S = n > 1$ at every $1/n$-th pulse, s pulses are added to the normal consumption for the following measurement during Δt.

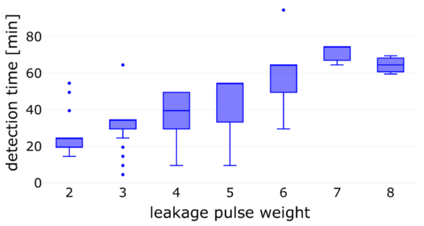

Fig. 2. Time until detection for different pulse severity and strength calculated on artificial data.

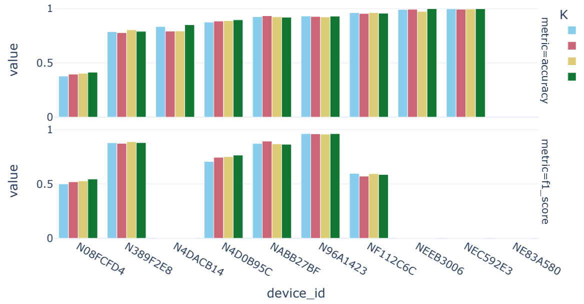

Fig. 3. Accuracy (top) and F1-score (bottom) of the leakage detection on the edge for all devices compared to the cloud solution. The color of the bars indicates the artificial temperature K.

With an overall accuracy of more than 93% and a F1-score of 90% over all samples, our edge algorithm performs very well even on 30 s data granularity. Further, the detection time (Fig. 2) on the edge ranges from less than 30 min from leakages with $S = 2$ to about one hour for leakages with lower severity. This fulfills the requirements of leakage detection of 1-3 hours as defined above.

2) Real-world data: For the evaluation of the leakage detection performance on the real-world data, we use the cloud-based leakage detection as ground truth. We expect our accuracy to be slightly lower, as we might detect leakages earlier. Hence, the ground truth does not yet indicate a leakage

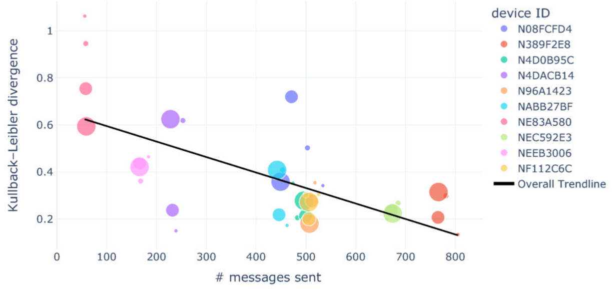

Fig. 4. Kullback-Leibler divergence for all devices given by the different colors. The size of the markers indicates the artificial temperature K.

though it has already started. In Fig. 3, we show the accuracy and the F1-score. For 80% of devices, the accuracy is higher than 0.8, while for one of the remaining devices (*NE83A580*), the ground truth data is missing. Further, the F1-score, the ratio of correctly detected instances of leakages, are similarly high as the accuracy. Note that the devices with a F1-score of zero are those devices that have (quasi) no leakages (c.f. Table I). For both devices (*NEC592E3*, *NEEB3006*), we only see a leakage in the first 5 minutes of the full data set which rather indicates a misclassification due to a cold start problem. The devices with lower F1-score, namely *N08FCFD4* and *NF112C6C*, are a school and a sports facility, respectively. By analysing the data in more detail, the following factors lead to a reduced accuracy and F1-score: i.e. first, for *N08FCFD4*, the average leakage is at 0.049 pulses, hence every twentieth pulse indicates a leakage. With the current threshold, this leakage is too weak to be detected reliably. However, for such a weak leakage, which is probably a dripping water tab, the timely detection is less crucial. For the latter, device *NF112C6C*, we can see that at the time when the actual leakage begins, the leakage detection at the edge triggers a warning almost 24 hours earlier than the cloud algorithm, which explains the reduced and in this case misleading F1-score.

Additionally, in order to ensure that it is not possible to extract privacy-sensitive information from the number of message that are sent, we calculate the Kullback-Leibler (KL) divergence $D_{KL}(P||Q)$ [12]. We chose the KL divergence as it is usually used in adversarial Neyman–Pearson tests for identifying distribution differences [13]. For P, we derive the discrete distribution from a sample of random variables drawn from the truncated exponential distribution as given in equation (1) derived from as many samples as messages were sent (for each device and temperature) with $T^\star = T$. Q is the discrete distribution derived from the random sending times when applying our algorithm. We use it as a measure of information gained when approximating the truncated exponential distribution P with the out-of-equilibrium distribution Q that enables early leakage warnings.

In Fig. 4 the divergence is shown against the number of send messages for all devices (color) and for different artificial temperatures (size). There is no clear correlation between the artificial temperature and the KL-divergence and hence does not change anything in the information that can be extracted from the distribution. On the contrary, the KL is clearly related to the number of messages being sent. Hence, over long run times of the algorithm, the KL divergence decreases such that no private information can be extracted from the distribution of messages sending times.

VI. Conclusion and Next Steps

We introduced an on-the-edge water leakage detection approach that addresses three contradicting requirements: (i) an overall reduced number of messages in order to extend the device's battery lifetime, (ii) early-alarming in case of a detected leakage on the edge, and (iii) a privacy-ensuring message sending schema. The approach is based on lightweight compression performed on the edge in order to timely detect leakages and on random messaging for privacy protection. We evaluated it against artificial as well as against real-world data from devices installed in different types of buildings, such as private households and public buildings. In both cases, we observe a high leakage detection accuracy as well as a timely detection of the leakage, fulfilling the industrial requirements.

As next steps, we plan to further improve our approach by considering building-type specific leakage detection. Additionally, we will analyse our approach on pipe bursts, which show a very specific pattern. From our initial results, we see already that we receive reliable warnings also for those. Further, we will perform an on-the-edge battery lifetime study in order to give a realistic estimation when applying our approach in production.

Acknowledgment

The work in this paper results from MIRAI, a project labelled by ITEA3 under project no 19034, with funding support from Innoviris Brussels, Belgium.

References

[1] H. Fuentes and D. Mauricio, “Smart water consumption measurement system for houses using iot and cloud computing,” *Environmental Monitoring and Assessment*, vol. 192, no. 9, pp. 1–16, 2020. doi: 10.1007/s10661-020-08535-4

[2] M. Fagiani, S. Squartini, L. Gabrielli, M. Severini, and F. Piazza, “A statistical framework for automatic leakage detection in smart water and gas grids,” *Energies*, vol. 9, no. 9, p. 665, 2016. doi: 10.3390/en9090665

[3] S. Alvisi, F. Casellato, M. Franchini, M. Govoni, C. Luciani, F. Poltronieri, G. Riberto, C. Stefanelli, and M. Tortonesi, “Wireless middleware solutions for smart water metering,” *Sensors*, vol. 19, no. 8, p. 1853, 2019. doi: 10.3390/s19081853

[4] S. Kartakis, W. Yu, R. Akhavan, and J. A. McCann, “Adaptive edge analytics for distributed networked control of water systems,” in *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 2016. doi: 10.1109/IoTDI.2015.34 pp. 72–82.

[5] J. Kraus and V. Bubla, “Optimal methods for data storage in performance measuring and monitoring devices.” in *Proceedings of electronic power engineering conference*, 01 2008. ISBN 9788021436503 pp. 131–133.

[6] T. Britton, G. Cole, R. Stewart, and D. Wiskar, “Remote diagnosis of leakage in residential households,” *Journal of Australian Water Association*, vol. 35, no. 6, pp. 89–93, 2008.

[7] A. Apostolico and A. Fraenkel, “Robust transmission of unbounded strings using fibonacci representations,” *IEEE Transactions on Information Theory*, vol. 33, no. 2, pp. 238–245, 1987. doi: 10.1109/TIT.1987.1057284

[8] S. T. Klein and M. K. Ben-Nissan, “On the usefulness of fibonacci compression codes,” *The Computer Journal*, vol. 53, no. 6, pp. 701–716, 2010. doi: 10.1093/comjnl/bxp046

[9] A. Robinson and C. Cherry, “Results of a prototype television bandwidth compression scheme,” *Proceedings of the IEEE*, vol. 55, no. 3, pp. 356–364, 1967. doi: 10.1109/PROC.1967.5493

[10] C. Beckel, L. Sadamori, and S. Santini, “Automatic socio-economic classification of households using electricity consumption data,” in *Proceedings of the fourth international conference on Future energy systems*, 2013. doi: 10.1145/2487166.2487175 pp. 75–86.

[11] C. Kittel, *Elementary statistical physics*. Courier Corporation, 2004.

[12] S. Kullback and R. A. Leibler, “On information and sufficiency,” *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951. doi: 10.1214/aoms/1177729694. [Online]. Available: http://www.jstor.org/stable/2236703

[13] Z. Li, T. J. Oechtering, and D. Gündüz, “Privacy against a hypothesis testing adversary,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1567–1581, 2018. doi: 10.1109/TIFS.2018.2882343