Measuring the Carbon Footprint of Cryptographic Privacy-Enhancing Technologies

Marc Damie[1,2]*, Mihai Pop[1], and Merijn Posthuma[1]

[1] University of Twente, The Netherlands
[2] Inria, France

Abstract. Privacy-enhancing technologies (PETs) have attracted significant attention in response to privacy regulations, driving the development of applications that prioritize user data protection. At the same time, the information and communication technology (ICT) sector faces growing pressure to reduce its environmental footprint, particularly its carbon emissions. While numerous studies have assessed the energy footprint of various ICT applications, the environmental footprint of cryptographic PETs remains largely unexplored.

Our work addresses this gap by proposing a standardized methodology for evaluating the carbon footprint of PETs. To demonstrate this methodology, we focus on PETs supporting client-server applications as they are the simplest to deploy. In particular, we measure the energy consumption and carbon footprint increase induced by five cryptographic PETs (compared to their non-private equivalent): HTTPS web browsing, encrypted machine learning (ML) inference, encrypted ML training, encrypted databases, and encrypted emails. Our findings reveal significant variability in carbon footprint increases, ranging from a twofold increase in HTTPS web browsing to a 100,000-fold increase in encrypted ML.

Our study provides essential data to help decision-makers assess privacy-carbon trade-offs in such applications. Finally, we outline key research directions for developing PETs that balance strong privacy protection with environmental sustainability.

Keywords: Privacy-Enhancing Technologies · Carbon Footprint · Cryptography· Encrypted ML · Encrypted Databases · HTTPS.

1 Introduction

Over the last ten years, awareness about privacy issues has significantly increased. Legislations such as the European GDPR (General Data Protection Regulation) marked a turning point by incentivizing practitioners to develop applications considered as "private by design." Pushed by these legal incentives, the research community has proposed many novel privacy-enhancing technologies (PETs); enabling to implement existing applications without requiring the user to reveal their private data.

* Corresponding author: m.f.d.damie@utwente.nl

However, information and communication technologies (ICTs) are also coping with another major challenge: reducing their environmental footprint; in particular their carbon emissions. In their fight against climate change, many governments are willing to reduce their carbon footprint, and all industries (including ICTs) attempt to contribute to the carbon emission reductions.

Several existing works started exploring this problem either by estimating the carbon footprint of the ICT industry [21] or by measuring the footprint of specific applications. Measurements are essential to identify the most energy-efficient technologies in order to fulfil carbon emission reduction goals.

Related works Existing works have measured the environmental footprint of various ICTs: online advertising [2], network communications [15], video streaming [1], video calls [6], Machine Learning (ML) [17,23,41], and blockchain [3,42]. However, there has been limited research on the impact of PETs.

On the one hand, several works [33,36] have studied the energy consumption of the TLS protocol used to secure various web protocols (including HTTP). This represents important related works as TLS is a form of "soft" PET (i.e., a PET providing data security and processing data with consent [11]).

On the other hand, various ML paradigms have been studied under the lens of carbon footprint, including differentially private ML [35,38] and Federated ML [19,40]; two paradigms related to privacy-preserving ML. Our work is complementary to these research works because we notably measure the footprint of cryptographic techniques applied to privacy-preserving ML.

Gap in the literature Despite these preliminary works, the debate about environmental sustainability is absent from the cryptography community. This absence is particularly problematic because cryptographic PETs induce computation and communication overheads compared to their "non-private" equivalents. While privacy could be seen as a value to protect at all costs, the emission reduction goals may require to design PETs offering the best trade-off between privacy and carbon emissions.

Unfortunately, the literature provides no evaluation of the environmental footprint of cryptographic PETs. Such measurements have become essential to find the best trade-off between privacy and carbon emissions.

Our contributions We present a **standard methodology** to analyze the footprint of a PET. In particular, we detail a taxonomy of the possible overheads induced by a PET. This methodology enables to answer two key questions: (1) What is the carbon footprint of a given cryptographic PET? (2) What is the relative increase compared to a non-private equivalent?

To demonstrate this methodology, we **measure the energy consumption and carbon footprint** of five cryptographic PETs: HTTPS, encrypted ML inference, encrypted ML training, encrypted databases, and encrypted emails. We focus on PETs supporting client-server applications, because they are the easiest to deploy. In particular, we **highlight the relative carbon footprint difference** between the PET and its non-private equivalent (i.e., the carbon

footprint of the privacy enhancement). We show highly variable carbon footprint increases from a $\times 2$ increase in HTTPS to a $\times 100,000$ in encrypted ML.

Finally, we discuss **promising research directions** to build sustainable and privacy-enhancing technologies.

Our goal is not to categorize PETs between environmentally acceptable and unacceptable. To fulfil carbon emission goals, it is not mandatory to reduce the carbon emissions of all technologies: some highly energy-consuming technologies can be considered "worth the emissions" because their service is essential. Our work simply provides orders of magnitude enabling decision makers to assess the privacy-carbon trade-offs of specific services. Such figures have become essential in the public debate as press articles rely on such estimations and measurements to provide high-level perspectives on sustainability issues [18].

2 Measuring the carbon footprint of a PET

Life-Cycle Assessment (LCA) is a standard methodology [22] to measure the environmental footprint associated with all stages of a product's life. Such stages can include anything from the manufacturing to the recycling. This approach enables to estimate the energy consumption, carbon emissions, and resource consumption in a comprehensive manner.

Taking inspiration from Schmidt et al. [41] who introduced carbon footprint measurement to the ML community, this section applies the LCA methodology to PETs and present simple analysis tools for PET researchers.

2.1 Background: Life-Cycle Assessment

An LCA requires **defining a scope** based on an analysis goal; this will specify what is included in and excluded from the analysis. Let us assume that we want to compare the footprint of two objects. If the two objects use the same materials and end-of-life processes, the material extraction and the end of life are not necessary in the analysis. Indeed, these stages affect equally the two objects, so they will not change the comparison (i.e., our objective).

The analysis goal also defines an analysis metric which quantifies the impact of a product. The two most common metrics are the energy consumption (in kWh) and the carbon emissions (in kg eq. CO2) [41]. However, some analyses also quantify other variables such as water consumption [5].

LCA and digital services While LCA focuses on physical objects, it is possible to extend this approach to digital services. However, LCA on such services focuses mostly on the service usage. For example, for an ML service, an LCA studies the impacts of the model training [23,30,41], and of the inference [14].

One may wonder why the hardware is not always included in the setup. Even though hardware is a significant part of the ICT footprint [21], the hardware-related costs are irrelevant in the scope of these works. These works [23,41] (like ours) want to identify the most energy-efficient services and execute them on a

standard hardware. In other words, the hardware-related footprint is excluded because they want to minimize the footprint of the service usage.

Like most works on the carbon footprint of ICTs [17,23,33,35,36,38,40,41], our LCA focuses on the service usage. Therefore, our experiments quantify the impact in terms of **carbon footprint and energy consumption**.

2.2 Taxonomy of carbon footprint overhead in PET

Our goal is to measure the "footprint overhead" induced by PET usage; i.e., the footprint increase between a PET and a non-private equivalent. This footprint increase can have several origins in PETs. To compare the carbon footprint of PETs, we distinguish four types of overhead:

- **Computational overhead**: any additional computation (e.g., encryption).
- **Communication overhead**: any additional communication (e.g., if the ciphertext is larger than the plaintext).
- **Infrastructure overhead**: any server duplication (e.g., like the existence of two non-colluding servers in privacy-preserving telemetry [9]).
- **Hardware overhead**: any PET-specific hardware requirement (e.g., TEE).

2.3 Our scope: client-server applications

As there exists plenty of PETs, we demonstrate our methodology on a specific subset and leave the rest of them for future works. Our work focuses on PETs in a client-server setup because they are the easiest to deploy. This simplicity makes such PETs attractive to companies providing "Software as a Service" because they simply need to update their code and do not require major infrastructure changes. For example, encrypted databases (such as MongoDB Queryable Encryption) fit this category because this PET only requires updating the software.

Thus, we compare the computation costs of a client-server PET deployment to the non-private equivalent. More precisely, we highlight the relative footprint increase due to the privacy enhancement. This focus on client-server setups implies that the studied PETs induce no infrastructure overhead because, like the non-private equivalent, such PETs only require a client and a server.

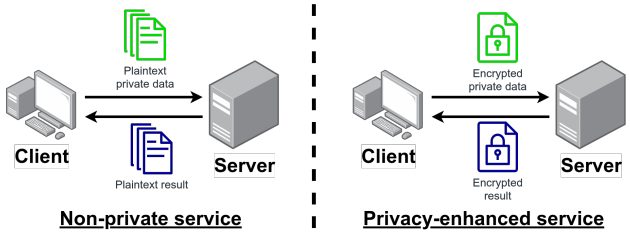


Fig. 1: Our scope: non-private vs. privacy-enhanced client-server technology

Moreover, we exclude communication overheads from our scope because no existing work enables quantifying *appropriately* this overhead. Indeed, measuring the footprint of network communications is still an open research problem. No work has yet estimated the complete footprint of a client-server communications because it requires computing the amortized energy consumption of all network devices (e.g., routers, switches, antennas, wires, etc.) used to transmit a bit of information. Ficher et al. [15] started exploring this question, but they studied only the communication between two servers in the same network backbone. Further research is still necessary to obtain a general result.

Anyway, client-server PETs usually induce minor communication overheads. For example, in HTTPS web browsing, the size of the ciphertext sent to the server is comparable in size to the plaintext (plus a one-time secret-key exchange). Their communication overhead is anecdotal compared to communication overheads in "communication-intensive PETs" such as multiparty computations (MPC).

Finally, we leave for future works PETs based on cryptographic hardware (e.g., Trusted Execution Environments [39]). Their need for a dedicated hardware calls for a fine-grained analysis of this "hardware overhead."

In summary, **due to the properties of client-server PETs, our work only need to focus on the computational overhead**. This scope already fits many PETs and our experiments covers 5 of them. Future works would be necessary to assess the carbon footprint of all possible PETs. Our research provides a clear framework supporting such future research and demonstrate its practicality on the PETs analyzed in Section 3.

2.4 Software-based measurement

Our scope requires measuring the energy consumption induced by a software. A natural approach is to use a power meter. While reliable, such hardware-based approaches are inconvenient as they limit the reproduction.

On the contrary, recent works [24,26,41] have promoted software-based approaches to measure the energy consumption. Software-based measurements are valuable as they simplify the reproduction of an experiment: they simply require a package installation. In particular, Khan et al. [26] showed that Intel's Running Average Power Limit (RAPL) was a powerful tool to measure the power consumption of a machine. RAPL-based measurement consists in polling the RAPL interface every x seconds and extrapolate the consumption based on these discrete measurements. Such measurements were notably used in ML research [41]. Recently, Jay et al. [24] highlighted that RAPL induces a low overhead.

Considering the advantages of RAPL, like related ML works [41], we rely on this software-based approach to measure energy consumption. Schmidt et al. [41] extended this approach to include GPU and RAM consumption.

In addition to energy consumption, our scope also expects carbon emission measurement. To compute the carbon emissions, one needs to multiply the energy consumption to the "carbon intensity" of the country in which the server/client is located. The carbon intensity corresponds to the amount of greenhouse gas emitted per kWh of electricity produced. This information is directly

available in some public databases [13]. In other words, the energy consumption and the energy emissions are perfectly correlated. This carbon emission estimation is commonly used in ML research [23,41].

2.5 Is the runtime a good proxy for the carbon footprint?

As research papers typically benchmark the runtime of their algorithms, one may want to extrapolate the carbon footprint based on such measurements. While runtime is correlated to energy consumption, other parameters influence it. Thus, we cannot trivially deduce the energy consumption based on the runtime.

To reduce their runtime, research works (including in PETs [27,32]) parallelize their computation across multiple cores. When parallelized, the energy consumption of an algorithm is proportional to the number of cores in use. Thus, a runtime can be divided by 4, 16, or even 32 via parallelization, but the energy consumption would remain similar because spread over several cores. Similarly, powerful CPUs and GPUs can also reduce the runtime, but induce a higher energy consumption than smaller chips.

This observation shows that specialized energy measurement techniques (such as RAPL [26]) are essential to quantify the carbon footprint and cannot be obtained using simpler methods such as runtime measurement. For example, RAPL-based measurements accounts for the energy consumption of all cores.

In conclusion, we provide a complementary perspective to existing PET papers because we report carbon and energy measurements that *cannot be precisely extrapolated from existing experimental results*. Our work echoes with recent ML works [23] that promoted a systematic reporting of carbon footprint and energy consumption in ML papers (in addition to existing runtime measurements).

2.6 Experimental setup

We execute our experiments on a dedicated server with 32 GB, an Intel Xeon-E3 1245 v5, and no GPU. We run the client and the server on the same machine.

As the hardware can slightly change the energy consumption of an algorithm, one may wonder whether we should execute our experiment on different devices. However, remember that we mainly want to measure the relative footprint difference between private and non-private deployments. A standard hardware change would influence the absolute values, but the relative difference should not be significantly changed. This LCA on a single hardware platform aligns with common practices in studies performing LCA on ML algorithms [38,19].

The only hardware that could significantly influence a PET benchmark is cryptographic accelerators. We leave them out-of-scope because a dedicated analysis is necessary to estimate the production cost of this hardware *only needed* in the private deployment. Section 4.1 further discusses the use of such hardware.

We use the software CodeCarbon [41] to measure the energy consumption and infer the subsequent carbon emissions. This software uses a RAPL-based measurement, and measures RAM, CPU, and GPU consumption.

We configured CodeCarbon to poll the energy consumption every 1 ms (like in [24]). Moreover, we chose the Netherlands as reference country to compute the carbon emissions. The Netherlands have a quite "average carbon intensity" within Europe (i.e., 268 g eq. CO2 per kWh in 2023) compared to the low carbon intensity of France (56 g eq. CO2 per kWh) and the high carbon intensity of Poland (662 g eq. CO2 per kWh). However, the choice of a country does not matter too much: we want to measure the relative increase. Reproducing the experiment in another country would change the absolute carbon emission but not the relative difference.

As explained above and illustrated in Figure 2, the energy and carbon emissions are then perfectly correlated in our experiments. Thus, some figures (as Figures 4a and 5a) only report the energy consumption in order to comply with space limitations. The reader can easily extrapolate the carbon emissions of a PET in any country by multiplying the energy consumption (reported in our figures) with the carbon intensity of the desired country [13].

3 Experimental results

This section measures the **relative difference** between the footprint of different PETs and the footprint of their non-private equivalent. Our measurement aggregates the computational costs of the client and the server.

Our experiments systematically use the most default configuration proposed in the documentation of the tested software. Thus, the measurements reported in our paper correspond to the expected default behavior. For implementation details, we refer to our **publicly available codebase**: `https://github.com/MarcTOK/privacy-carbon-experiments`

3.1 Encrypted Machine Learning inference

With the increasing use of ML, there is a growing concern regarding the use of personal data in these systems. To address this problem, many works [12,34] have proposed privacy-preserving ML protocols to support these applications while preserving data privacy. Among all ML operations, ML inference (i.e., evaluating data on a trained ML model) is a particularly important operation for practitioners. Indeed, many companies have trained a powerful model (e.g., a Large Language Model) and provide "Inference as a Service": customers send input data and the company returns the model output.

However, traditional "Inference as a Service" requires sharing personal data in plaintext with the service provider. To enhance privacy, researchers have designed encrypted ML frameworks [8] (based on Fully Homomorphic Encryption) to perform ML operations (in particular inference) under encryption.

Software libraries. Our goal is to measure the energy consumption of an encrypted inference compared to a classic plaintext inference. We use the software library Concrete ML developed by the company Zama. This library is stable and promoted by the company for real-world use cases. We compare the encrypted

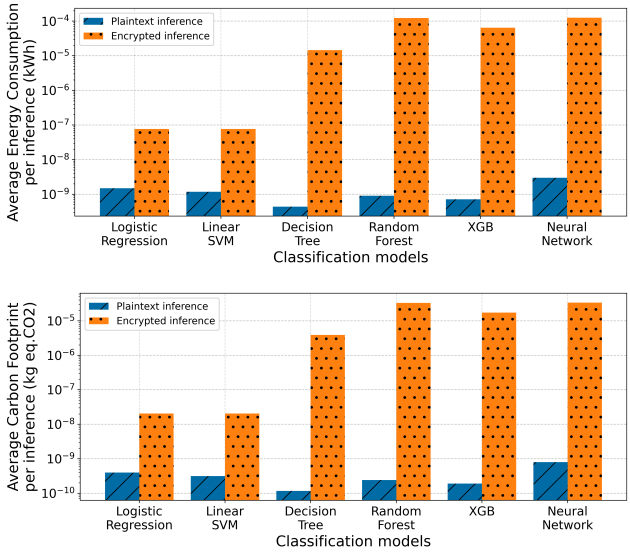


Fig. 2: Average energy consumption and carbon emissions of encrypted and plaintext ML inference using various classification models (100 samples, 30 features).

inference provided by Concrete ML to the plaintext ML inference performed by Scikit Learn (a popular ML framework). Note that Scikit Learn is also used as baseline by Zama in their accuracy benchmarks.

Data. We use Scikit Learn to generate synthetic datasets with varying number of features. In ML, a sample is a data point included in a dataset, and the features are the dimensions/characteristics of a data point.

Results. Figures 2 and 3 presents measurements for various classification and regression models. These results are average measurements over 100 samples; each sample having 30 features. On linear models (e.g., Logistic regression), the encrypted inference is 100× more expensive than the plaintext inference. On tree models (Decision Tree, Random Forest, and XGB), the encrypted inference is at least $100,000\times$ more expensive than the plaintext inference. Encrypted inference on neural networks is also $100,000\times$ more expensive than plaintext.

To better understand this overhead, Figure 4a studies the influence of the number of features on the footprint of three popular classification models: logistic regression, random forest, and neural networks. We observe that the scaling of the encrypted inferences does not match perfectly the scaling of the plaintext inferences. First, the encrypted logistic regression footprint increases relatively faster than the cost of the plaintext inference. Second, the encrypted neural network seems to have the same scaling pattern as the encrypted logistic regression. The encrypted neural network does not have results for more than 200 features because we stopped inferences taking more than several hours (to limit the energy consumption of our experiments). Third, the encrypted random for-

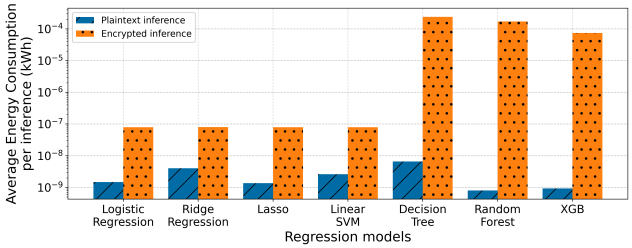
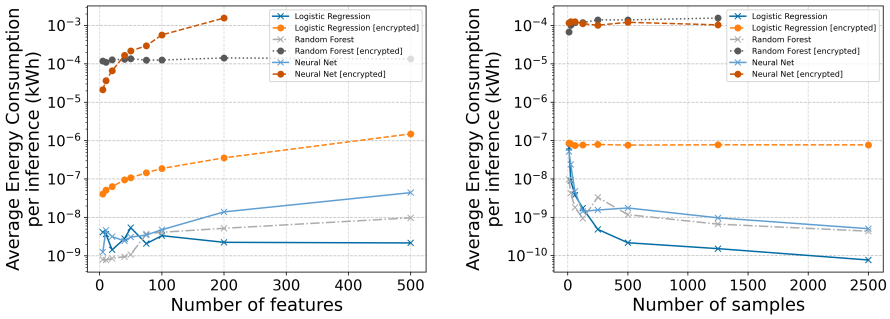


Fig. 3: Average energy consumption of encrypted and plaintext ML inference using various regression models (100 samples, 30 features).



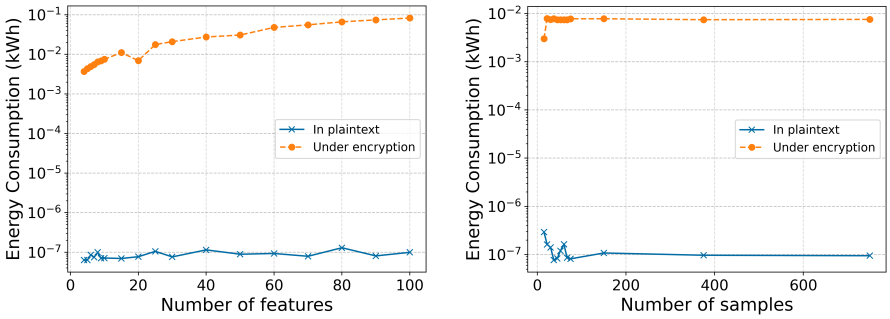
(a) Varying nb. of features / 100 samples (b) Varying nb. of samples / 30 features

Fig. 4: Average energy consumption of encrypted and plaintext ML inference for varying number of features and samples.

est footprint is approximately constant. This behavior seems to be an artifact coming from the fact that Concrete ML transforms decision trees into matrix multiplications (inducing significant constant costs).

Figure 4b illustrates the footprint in function of the number of samples. On the one hand, we see that plaintext algorithms are optimized to process batches of data points: the average cost is amortized when the number of samples grows. On the other hand, we observe on the logistic regression that the encrypted inference does not provide a similar amortization. Like on Figure 4a, we stopped experiments requiring several hours (which explains the smaller number of results for the random forest and neural network).

Discussions. Our experiments show that encrypted ML inference induces significant overheads. However, this PET is still relatively recent, so there is room for optimization. For example, Ko et al. [28] recently presented an encrypted XGB 100× more efficient than Concrete ML's XGB. This improvement resonates with Figure 4a that identified a major constant factor for encrypted Random Forest (i.e., a tree-based model related to XGB). Our benchmark could be re-executed once Concrete ML integrates improved algorithms.



(a) Varying nb. of features / 300 samples (b) Varying nb. of samples / 10 features

Fig. 5: Energy consumption of encrypted and plaintext ML training of a logistic regression for varying numbers of features and samples

3.2 Encrypted Machine Learning training

Next to inference, ML training is the other operation attracting a lot of privacy concerns. Like for inference, researchers have designed frameworks [12,16] to execute these algorithms directly on encrypted data. These frameworks usually rely on the same cryptographic primitives as encrypted inference.

Software libraries. Even though the Concrete ML library focuses on encrypted inference, it also implements a few training algorithms. In particular, it enables training a logistic regression under encryption. We then compare this encrypted training to the logistic regression of Scikit Learn.

Data. Like for inference, we use Scikit Learn to generate synthetic data with a varying number of samples and features.

Results. Figure 5a compares the encrypted and plaintext training for a varying number of features. The encrypted training is at least $100,000\times$ more expensive and its costs increase faster with the number of features.

Contrary to inference, the logistic regression training is not influenced by the number of samples. Training algorithms are iterative and process a fixed-sized batch during each iteration. Thus, their cost only depends on the batch size and number of iterations. Figure 5b confirms this behavior experimentally.

Discussions. Like secure ML inference, secure training induces massive footprint increase. Similarly, there is room for optimizations.

However, other (non-cryptographic) approaches to privacy-preserving ML training exist. For example, de Reus et al. [38] studied the carbon footprint of synthetic data generation in ML context. Synthetic data generation [25] is a pre-processing step used in privacy-preserving statistics and ML: instead of encrypting its private data, the data owner generates synthetic data based on its data and can transmit the synthetic data in plaintext. In this paradigm, the server can then process plaintext data without privacy issues as it has only access

to synthetic data (and not the initial private data). Some techniques such as [25] ensure that the synthetic data does not leak private information.

De Reus et al. [38] showed that their studied synthetic data generation requires 1 Wh to generate a synthetic dataset based on the adult dataset (14 features), and 0.01 Wh to train the logistic regression. They also used a RAPL-based energy measurement, so our results are comparable. In comparison, Figure 5a shows that the encrypted ML training requires 10× more energy for a training on a similar dataset.

This comparison emphasizes that the PET choice has a major impact on the carbon footprint. It also highlights an interesting future work: comparing all privacy-preserving ML paradigms to identify the most energy-efficient one.

3.3 Encrypted databases

With the rise of cloud services, vast amounts of personal data are stored on outsourced databases, raising privacy concerns since providers may not be fully trusted. To solve this issue, researchers introduced searchable encryption [10], a database model that enables encrypted data storage and querying while preventing the provider from accessing data or query content.

Software libraries. This experiment[3] uses SWiSSSE [20] as baseline for encrypted database. SWiSSSE is an encrypted database system comparable to Redis (a traditional database specialized in key-value storage).

Gui et al. [20] already compared the runtime of SWiSSSE to Redis. We reproduce their benchmark using new metrics: energy and carbon.

Data. Like in [20], we populate the database using the Enron email dataset.

Results. Figure 6 shows the results of this benchmark for varying database sizes. We observe that the footprint of the encrypted database is nearly ten times higher than the footprint of the plaintext database. Moreover, the footprint of the encrypted database increases slightly faster in function of the database size.

Discussions. Our benchmark relies on SWiSSSE, which is a research prototype. Ideally, we would like to reproduce the results on professional products such as the "Queryable Encryption" plugin available in MongoDB. Unfortunately, part of this plugin requires a premium license; limiting its access and hindering reproducibility. We demonstrated our methodology on SWiSSSE, and leave for future work the evaluation of MongoDB Queryable Encryption.

3.4 HTTP vs. HTTPS

HTTP (HyperText Transfer Protocol) is the foundational protocol used for data exchanges on the web. It operates as a plaintext protocol; making the transmitted data susceptible to eavesdropping and tampering. In contrast, HTTPS integrates HTTP with the Transport Layer Security (TLS), encrypting data to

[3] Due to hardware issues (unrelated to the experiment), we executed this experiment on a MacOS machine while the other experiments are executed on a Debian server.

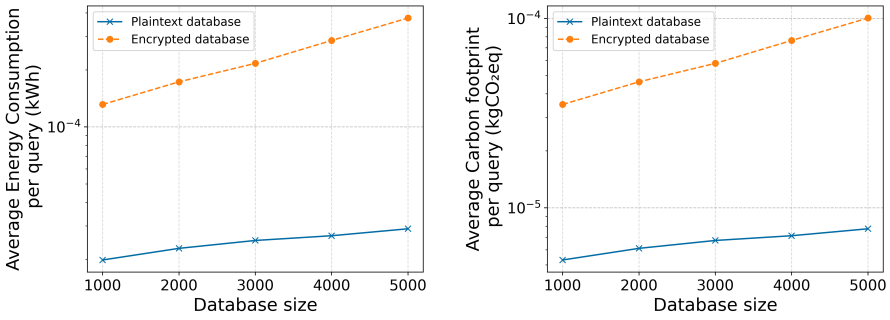


Fig. 6: Average energy consumption and carbon emissions of encrypted and plaintext queries (SWiSSSE vs. Redis) for varying database sizes (1000 queries).

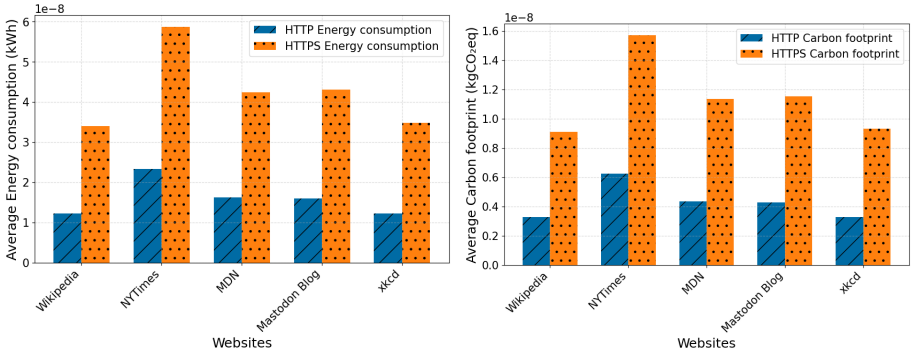


Fig. 7: Average energy consumption and carbon emissions of HTTP and HTTPS requests on five different websites (1000 requests).

ensure confidentiality and integrity. HTTPS protects sensitive data, such as login credentials and payment information, from being intercepted or altered.

Software libraries. We use NGinx as web server with TLS 1.3 and implemented a basic Python web client using the `requests` library. We compute the average energy consumption and carbon emissions over 1000 Web requests.

Data. We run this experiment on five different websites: Wikipedia (Simplified English), New York Times, MDN, Mastodon Technical Blog, and xkcd. These websites are quite diverse: Wikipedia is an encyclopedia, New York Times a media, MDN a web developer documentation, Mastodon Blog is a blog, and xkcd is a minimal website publishing amusing comic strips. Wikipedia source files are publicly available, so we downloaded the "simplified English" archive from April 2007. We used the tool HTTrack to automatically download the static files (i.e., HTML, CSS, JS, and images) of the other websites.

Results. Figure 7 represents the average energy consumption and carbon footprint over 1000 HTTP(S) requests. The relative footprint increase ranges from 152% for NYTimes to 182% for xkcd.

Unfortunately, we cannot compare our results to existing works on the energy consumption of TLS [33,36]. While we analyze the energy consumption to estimate the environmental footprint of HTTPS, these works had a completely different goal: estimating the impact of HTTPS on a smartphone battery life. Thus, these works have only measured client-side energy consumption.

Moreover, Miranda et al. [33] focused on the energy consumption of TLS only; they do not provide a relative difference between HTTP and HTTPS. Finally, Naylor et al. [36] focused on specific Web applications (e.g., Youtube); contrary to our experiments that considered an easily reproducible setup: Python web client and static HTML files. Naylor et al. [36] also integrated some network-related costs (e.g., WiFi communications). These major divergences between our experimental setup and [36] prevent any naive comparison.

Discussions. Our experiments only simulate static websites, and exclude any possible back-end operations (e.g., database interactions), even though such operations are common in Web applications. To assess the footprint of a Web application, an experiment would include both the cost related to HTTP(S) and the back-end costs. However, this is not our goal: we only want to compare HTTP to HTTPS. Hence, we exclude any backend work to focus solely on the footprint of HTTP data transfer.

3.5 Encrypted emails

Protecting personal communications has long been a key objective for cryptographers. Over the decades, various solutions have emerged, ranging from encrypted emails to secure messaging apps like Signal. In this work, we focus on email encryption, as it is the most mature technology. Specifically, we examine the footprint of GPG encryption and signing, given that GPG is a widely used standard for securing email communication.

Software libraries. We use the GnuPG library (i.e., the reference open-source implementation for GPG). We benchmark 3 cipher suites: RSA, Elliptic Curve Cryptography (ECC), and ElGamal. Note that ElGamal only provides encryption so we combine it with DSA to sign messages. For each cipher suite, we used key sizes provided 128 bits of security [4].

Contrary to the other PETs, we do not have a non-private baseline to benchmark. Indeed, the non-private equivalent of email encryption is... no encryption (i.e., 0 cost). Thus, we present the encryption costs and put them into perspectives thanks to our previous experiments.

Data. We use the Enron email dataset to benchmark email encryption. We extract the 30109 emails contained in the sent mail folders of this dataset.

Results. Figure 8 presents the results. First, the encryption is systematically slightly more expensive than the signature for RSA and ECC. Second, RSA is slightly less expensive than ECC. Finally, ElGamal encryption is nearly ten times more expensive than RSA and ECC.

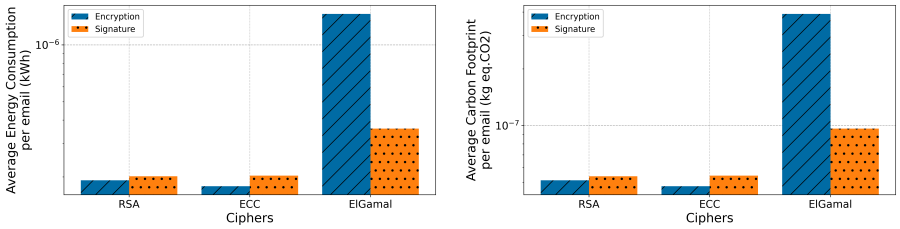


Fig. 8: Average energy consumption and carbon emissions of GPG email encryption and signing (30K emails).

For ElGamal, remember that we use DSA for signatures. The figure shows that DSA is about twice as expensive as RSA and ECC signatures.

To put these results into perspectives, we can compare them to secure ML inference. Based on our results, the footprint of an email encryption is then 1000× smaller than the footprint of an encrypted random forest inference.

Discussions. Our comparison shows comparable computational costs for RSA and ECC. However, ECC has another advantage over RSA: smaller key sizes. ECC could reduce (by a couple of kilobytes) the communication costs related to key and signature exchanges. As explained in Section 2.3, there is no consensus (yet) on the overall footprint of network communications, so we cannot include this factor in our comparison.

Finally, email encryption is becoming less and less popular; in favor of secure messaging apps that are more accessible. Such secure messaging systems rely heavily on symmetric-key cryptography, which is known to be more efficient than public-key cryptography. Secure messaging apps could then provide even smaller footprints than those of email encryption.

However, such messaging systems also require expensive protocols to set up communications. For example, half the operation costs of Signal Messenger is caused by the registration process [43]. An LCA of Signal should then include these expensive operations and not only the cheap symmetric-key encryption.

Our analysis of email encryption provides valuable baseline, but the analysis of complex messaging systems such as Signal represents a promising future work. Such analysis is particularly challenging as some protocols used by Signal require cryptographic hardware like secure enclaves whose environmental footprint is harder to estimate (as discussed in Section 4.1).

4 Orthogonal discussions

4.1 The hidden cost of cryptographic hardware

Energy consumption can be significantly reduced by using cryptographic accelerators [7]. These chips can perform cryptographic operations much more efficiently than standard CPUs and GPUs.

However, the LCA must include the manufacturing impact, as this specialized hardware adds a hardware overhead in our taxonomy. Since manufacturing accounts for most of the ICT sector's carbon footprint [21], using cryptographic accelerators does not necessarily reduce the overall carbon footprint.

Trusted Execution Environments (TEEs) [39] are another example of cryptographic hardware with an environmental cost that should not be ignored. TEEs are now widely used in efficient secure protocols because they allow operations on confidential data without relying on expensive primitives like homomorphic encryption. For example, Signal uses TEEs in its Contact Discovery protocol.

As their use grows, more manufacturers now include TEEs by default in their products. However, the added manufacturing cost still represents an overhead that needs to be estimated. Like cryptographic accelerators, estimating this cost is difficult, as the necessary data is rarely made public by production facilities. This gap then represents a promising and interesting future work.

4.2 Privacy-Carbon-Functionality trilemma

Our work initially examined the relationship between privacy and carbon emissions in an isolated manner. However, our experiments indirectly revealed a broader trilemma involving privacy, carbon emissions, and functionality.

The carbon footprint of cryptographic PETs is tied to their functionality; more complex functionalities generally require more computationally expensive cryptographic primitives, leading to higher emissions. This phenomenon is evident in Figure 2. While encryption always increases the footprint, the extent of this increase varies significantly across ML models. For instance, logistic regression exhibits a 100-fold increase, whereas encrypted neural networks result in a 100,000-fold increase. Consequently, simpler models like logistic regression lead to a smaller footprint.

Reducing functionality can therefore serve as an effective strategy to mitigate the overhead induced by the privacy enhancement.

4.3 Mitigating carbon footprint overhead via decentralization

Besides functionality, trust is another powerful leverage to optimize the privacy-carbon trade-off. The PET literature usually considers two edge cases: (1) a party trusted by all users requiring no PET and (2) a zero-trust world requiring (potentially) expensive PETs.

However, decentralized social media such as Mastodon introduced a new kind of trust assumption: decentralized and personalized trust. In these social media, each user can pick a specific server (that they trust to process their personal data), and then they can interact with any user (even from other servers) thanks to the protocol ActivityPub. Such decentralization is comparable to the decentralization of the email protocol.

We can formulate the Mastodon threat model as follows: each user trusts one specific server (among all possible servers), and they do not trust anyone

else. Such decentralized trust avoids expensive cryptographic operations (⇒lower carbon emissions) because each server can perform plaintext operations on the personal data of its users. For instance, on Peertube (a "decentralized Youtube"), each server can offer ML-based video recommendations to its users without relying on encrypted ML.

Decentralization "*à la* Mastodon" is then a promising direction to enhance privacy while avoiding significant carbon footprint overheads.

Mansoux and Roscam [31] presented this decentralized trust as "social approach of privacy", opposed to the technical approach adopted by cryptographic PETs. While this "social" approach provides weaker privacy guarantees, Lee and Wang [29] showed that privacy was a key factor driving the adoption of Mastodon. Thus, the weaker guarantees are not necessarily an issue for user acceptability (even among privacy-aware users).

5 Conclusion

Our work studied the environmental footprint of cryptographic PETs. We first introduced a standardized methodology for measuring the carbon footprint of PETs; demonstrating it on five cryptographic PETs. Our results highlight (PET-induced) carbon footprint increases ranging from $2\times$ to $100,000\times$. Our findings provide essential data to assist decision-makers assessing the privacy-carbon trade-offs inherent in different ICTs.

Runtime has always been a natural metric in PET papers, and recent works [37] also mention monetary costs on Amazon Web Services. Even though such costs are relatable to practitioners, they could be completed with energy consumption (and possibly carbon emission) measurements. While monetary costs are valuable to economical actors, the environmental footprint is highly informative for the society as a whole (e.g., journalists or policymakers).

Future directions Our work marks a first step in the carbon footprint analysis of PETs, but significant work is still necessary to analyze the footprint of all existing PETs (especially complex systems such as Signal Messenger or Tor).

Acknowledgments. This work was supported by the Netherlands Organization for Scientific Research (De Nederlandse Organisatie voor Wetenschappelijk Onderzoek) under NWO:SHARE project [CS.011].

A Large Language Model has been **carefully** used to reformulate some sentences and fix typos.

References

1. Afzal, S., Mehran, N., Ourimi, Z.A., Tashtarian, F., Amirpour, H., Prodan, R., Timmerer, C.: A Survey on Energy Consumption and Environmental Impact of Video Streaming (Jan 2024). https://doi.org/10.48550/arXiv.2401.09854, arXiv:2401.09854 [cs]

2. Albasir, A., Naik, K., Plourde, B., Goel, N.: Experimental study of energy and bandwidth costs of web advertisements on smartphones. In: 6th International Conference on Mobile Computing, Applications and Services. pp. 90–97 (Nov 2014). https://doi.org/10.4108/icst.mobicase.2014.257770
3. Bada, A.O., Damianou, A., Angelopoulos, C.M., Katos, V.: Towards a Green Blockchain: A Review of Consensus Mechanisms and their Energy Consumption. In: 2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS). pp. 503–511 (Jul 2021). https://doi.org/10.1109/DCOSS52077.2021.00083
4. Barker, E.: Recommendation for key management: part 1 - general. Tech. Rep. NIST SP 800-57pt1r5, National Institute of Standards and Technology (May 2020). https://doi.org/10.6028/NIST.SP.800-57pt1r5
5. Berger, M., Finkbeiner, M.: Water Footprinting: How to Address Water Use in Life Cycle Assessment? Sustainability **2**(4), 919–944 (Apr 2010). https://doi.org/10.3390/su2040919
6. Berthoud, F., Ficher, M.: Évaluation de l'empreinte carbone d'une visioconference entre deux utilisateurs du service rendez-vous. report, CNRS - EcoInfo (Mar 2022), pages: 60
7. Bisheh-Niasar, M., Azarderakhsh, R., Mozaffari-Kermani, M.: Cryptographic Accelerators for Digital Signature Based on Ed25519. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **29**(7), 1297–1305 (Jul 2021). https://doi.org/10.1109/TVLSI.2021.3077885
8. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast Fully Homomorphic Encryption Library (Jun 2019)
9. Corrigan-Gibbs, H., Boneh, D.: Prio: Private, Robust, and Scalable Computation of Aggregate Statistics. In: 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17). pp. 259–282. USENIX Association (Mar 2017)
10. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. In: Proceedings of the 13th ACM Conference on Computer and Communications Security. pp. 79–88. CCS '06 (2006). https://doi.org/10.1145/1180405.1180417
11. Deng, M., Wuyts, K., Scandariato, R., Preneel, B., Joosen, W.: A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. Requirements Engineering **16**(1), 3–32 (Mar 2011). https://doi.org/10.1007/s00766-010-0115-7
12. Diaa, A., Fenaux, L., Humphries, T., Dietz, M., Ebrahimianghazani, F., Kacsmar, B., Li, X., Lukas, N., Mahdavi, R.A., Oya, S., Amjadian, E., Kerschbaum, F.: Fast and Private Inference of Deep Neural Networks by Co-designing Activation Functions. In: 33rd USENIX Security Symposium (USENIX Security 24) (2024)
13. Ember, Institute, E.: Carbon intensity of electricity generation – ember and energy institute (2024), accessed: 2025-01-16
14. Faiz, A., Kaneda, S., Wang, R., Osi, R., Sharma, P., Chen, F., Jiang, L.: LLMCarbon: Modeling the end-to-end Carbon Footprint of Large Language Models (Jan 2024). https://doi.org/10.48550/arXiv.2309.14393
15. Ficher, M., Berthoud, F., Ligozat, A.L., Sigonneau, P., Wisslé, M., Tebbani, B.: Assessing the carbon footprint of the data transmission on a backbone network (Mar 2021)
16. Frery, J., Stoian, A., Bredehoft, R., Montero, L., Kherfallah, C., Chevallier-Mames, B., Meyre, A.: Privacy-Preserving Tree-Based Inference with TFHE. In: Mobile, Secure, and Programmable Networking. pp. 139–156. Springer Nature Switzerland (2024). https://doi.org/10.1007/978-3-031-52426-4_10

17. Georgiou, S., Kechagia, M., Sharma, T., Sarro, F., Zou, Y.: Green AI: do deep learning frameworks have different costs? In: Proceedings of the 44th International Conference on Software Engineering. pp. 1082–1094. ICSE '22, Association for Computing Machinery (Jul 2022). https://doi.org/10.1145/3510003.3510221
18. Griffiths, S.: Why your internet habits are not as clean as you think. BBC (2020), accessed: 2025-03-05
19. Guerra, E., Wilhelmi, F., Miozzo, M., Dini, P.: The cost of training machine learning models over distributed data sources. IEEE Open Journal of the Communications Society **4**, 1111–1126 (2023)
20. Gui, Z., Paterson, K.G., Patranabis, S., Warinschi, B.: SWiSSSE: System-Wide Security for Searchable Symmetric Encryption. Proceedings on Privacy Enhancing Technologies (2024)
21. Gupta, U., Kim, Y.G., Lee, S., Tse, J., Lee, H.H.S., Wei, G.Y., Brooks, D., Wu, C.J.: Chasing Carbon: The Elusive Environmental Footprint of Computing. IEEE Micro **42**(4), 37–47 (Jul 2022). https://doi.org/10.1109/MM.2022.3163226
22. Hauschild, M.Z., Rosenbaum, R.K., Olsen, S.I.: Life Cycle Assessment: Theory and Practice. Springer (Sep 2017)
23. Henderson, P., Hu, J., Romoff, J., Brunskill, E., Jurafsky, D., Pineau, J.: Towards the systematic reporting of the energy and carbon footprints of machine learning. Journal of Machine Learning Research **21**(248), 1–43 (2020)
24. Jay, M., Ostapenco, V., Lefevre, L., Trystram, D., Orgerie, A.C., Fichel, B.: An experimental comparison of software-based power meters: focus on CPU and GPU. In: 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid). pp. 106–118 (May 2023). https://doi.org/10.1109/CCGrid57682.2023.00020
25. Jordon, J., Yoon, J., Schaar, M.v.d.: PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees (Dec 2018)
26. Khan, K.N., Hirki, M., Niemi, T., Nurminen, J.K., Ou, Z.: RAPL in Action: Experiences in Using RAPL for Power Measurements. ACM Trans. Model. Perform. Eval. Comput. Syst. **3**(2), 9:1–9:26 (Mar 2018). https://doi.org/10.1145/3177754
27. Klemsa, J., Önen, M.: PARMESAN: Parallel ARithMEticS over ENcrypted data (2023)
28. Ko, R., Mahdavi, R.A., Yoon, B., Onizuka, M., Kerschbaum, F.: SilentWood: Private Inference Over Gradient-Boosting Decision Forests (Nov 2024). https://doi.org/10.48550/arXiv.2411.15494, arXiv:2411.15494
29. Lee, K., Wang, M.: Uses and Gratifications of Alternative Social Media: Why do people use Mastodon? (Mar 2023). https://doi.org/10.48550/arXiv.2303.01285
30. Luccioni, A.S., Viguier, S., Ligozat, A.L.: Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model. Journal of Machine Learning Research **24**(253), 1–15 (2023)
31. Mansoux, A., Roscam Abbing, R.: Seven theses on the fediverse and the becoming of FLOSS (2020), publisher: Institute for Network Cultures and Transmediale
32. Mazzone, F., Everts, M., Hahn, F., Peter, A.: Efficient Ranking, Order Statistics, and Sorting under CKKS (Dec 2024). https://doi.org/10.48550/arXiv.2412.15126
33. Miranda, P., Siekkinen, M., Waris, H.: TLS and energy consumption on a mobile device: A measurement study. In: 2011 IEEE Symposium on Computers and Communications (ISCC). pp. 983–989 (Jun 2011). https://doi.org/10.1109/ISCC.2011.5983970

34. Mohassel, P., Zhang, Y.: SecureML: A System for Scalable Privacy-Preserving Machine Learning. In: 2017 IEEE Symposium on Security and Privacy (SP). pp. 19–38 (May 2017). https://doi.org/10.1109/SP.2017.12
35. Naidu, R., Diddee, H., Mulay, A., Vardhan, A., Ramesh, K., Zamzam, A.: Towards Quantifying the Carbon Emissions of Differentially Private Machine Learning (2021)
36. Naylor, D., Finamore, A., Leontiadis, I., Grunenberger, Y., Mellia, M., Munafò, M., Papagiannaki, K., Steenkiste, P.: The Cost of the "S" in HTTPS. In: Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies. pp. 133–140. CoNEXT '14 (Dec 2014). https://doi.org/10.1145/2674005.2674991
37. Newman, Z., Servan-Schreiber, S., Devadas, S.: Spectrum: High-bandwidth Anonymous Broadcast. In: 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22). pp. 229–248. USENIX Association (Apr 2022)
38. de Reus, P., Oprescu, A., van Elsen, K.: Energy cost and machine learning accuracy impact of k-anonymisation and synthetic data techniques. In: 2023 International Conference on ICT for Sustainability (ICT4S). pp. 57–65 (Jun 2023). https://doi.org/10.1109/ICT4S58814.2023.00015
39. Sabt, M., Achemlal, M., Bouabdallah, A.: Trusted Execution Environment: What It is, and What It is Not. In: 2015 IEEE Trustcom/BigDataSE/ISPA. vol. 1, pp. 57–64 (Aug 2015). https://doi.org/10.1109/Trustcom.2015.357
40. Savazzi, S., Rampa, V., Kianoush, S., Bennis, M.: An Energy and Carbon Footprint Analysis of Distributed and Federated Learning. IEEE Transactions on Green Communications and Networking pp. 1–1 (2022). https://doi.org/10.1109/TGCN.2022.3186439
41. Schmidt, V., Goyal, K., Joshi, A., Feld, B., Conell, L., Laskaris, N., Blank, D., Wilson, J., Friedler, S., Luccioni, S.: CodeCarbon: Estimate and Track Carbon Emissions from Machine Learning Computing (2021). https://doi.org/10.5281/zenodo.4658424, publisher: Zenodo
42. Sedlmeir, J., Buhl, H.U., Fridgen, G., Keller, R.: The Energy Consumption of Blockchain Technology: Beyond Myth. Business & Information Systems Engineering **62**(6), 599–608 (Dec 2020). https://doi.org/10.1007/s12599-020-00656-x
43. Whittaker, M., Lund, J.: Privacy is Priceless, but Signal is Expensive (2023)