Eddie DeMars

Dr. Kapadia

CSCI-B 430

4 December 2022

An Exploration of Using Intel SGX for Remote Computing in Healthcare

1 Introduction

In the realm of computer security, the healthcare industry has some particularly unique problems. While not all of a patient's data is sensitive, a peculiarly large amount of it is. The average company is trying to protect things such as a user's social security number, credit card information, password hash, etc. However, healthcare companies often have extensive data about a given person, such as their medical history, address, or current medications. The information healthcare companies possess could allow malicious actors to commit identity theft or even fraud, for example, allowing uninsured persons to use others' information to obtain healthcare via their insurance, increasing the overall price of insurance [1].

To protect patients' data, the Health Insurance Portability and Accountability Act (HIPAA) was passed in 1996. It was created to improve the portability of health insurance for people between jobs and combat fraud, but it also helped facilitate the digitization of healthcare records and regulates how patient information may be used [2]. The regulations mandated by HIPAA for the protection of patients were split into two parts, the HIPAA Privacy Rule and the HIPAA Security Rule.

The Privacy Rule defines what type of information is protected by HIPAA, referred to as Protected Healthcare Information (PHI). PHI is defined as: "'individually identifiable health information' held or transmitted by a covered entity or its business associate, in any form or media, whether electronic, paper, or oral" [3]. Companies must receive permission from patients before sharing PHI for most purposes. The Privacy Rule does not disallow the use of de-identified health information, which does not provide a reasonable way to identify an individual.

The HIPAA Security Rule has many rules that define what is required of a company when it comes to the protection of patients' electronic PHI (e-PHI), however, there are four general rules all

companies must follow [4]. First, they must ensure the confidentiality, integrity, and availability (also known as the CIA triad, a well-known security concept) of all e-PHI. Second, they must protect against threats to the security of the e-PHI. Third, they must protect against unlawful uses of the e-PHI. And fourth, they must ensure compliance from their workforce towards the e-PHI [4]. The Security Rule recognizes that it governs companies of many different sizes and capabilities, and as such does not mandate any specific security measures, but instead requires each company to perform risk analysis as part of their security procedures and to act according to their means and expected threats [4].

Not only do healthcare companies have special data that must be protected, but they also often have limited resources to do so. This makes cloud/remote computing/Hardware as a Service (HaaS) particularly attractive. However, cloud computing has privacy and security flaws that must be overcome to be utilized with sensitive data. It is currently tricky to protect data in use, as at some point the data must be unencrypted for operations to be performed on it. There are a few ways to solve this. One solution is to use trusted execution environments (TEEs) [5].

TEEs provide a secure environment within a computer that cannot be accessed or changed by unauthorized users, even the operating system itself, allowing operations to be performed on the data without worrying about exposing it [5]. This paper will focus on Intel Software Guard Extensions (SGX), an instruction set for computers with certain Intel CPUs that allows for the creation of TEEs called enclaves [6]. SGX creates an encrypted area of memory that the CPU protects from anything else on the computer, including the operating system and hypervisor. The user can then upload encrypted data to the enclave, which will then decrypt it, perform the program, then encrypt the results and send them back. This can be used to fix many of the issues of using remote computing for healthcare [6].

2 Approach

The bulk of this paper will be located in section 3, which will briefly explain how Intel SGX functions, then move on to two examples of possible schemas for using SGX in healthcare remote computing, and finally, will then examine some drawbacks and issues that SGX faces, both in the healthcare industry and overall. Section 4 will be composed of thoughts on how research into SGX in the

healthcare industry should proceed, as well as some suggestions on how SGX could be used in healthcare. Section 5 will contain a summary and final conclusions on the subject, and finally, Section 6 will contain any sources referenced.

3 Survey of Existing Techniques

3.1 Explanation of Intel SGX

As Intel SGX is a rather complicated and unique technology, a more thorough explanation of its workings and applications is necessary for this paper. It should be noted that this explanation will be hypothetical, as some security concerns may make some of it not apply in reality. As stated previously, SGX is a set of instruction codes in some Intel CPUs that allows for the creation of TEEs called enclaves. These enclaves are protected from unauthorized users through two main methods [6]. First is the Memory Encryption Engine (MEE). The MEE is responsible for encrypting the enclaves' DRAM content. This is primarily for protection against physical attacks, as the MEE is in the processor's memory controller, at the edge of the on-chip memory hierarchy. As such, it is insufficient for protection against software attacks. To protect against software attacks, SGX's main defense is a series of memory checks [6]. This ensures that software may not access any memory that does not belong to it, including other enclaves. With these protections, no malicious actors should be able to access the data [6].

To utilize SGX for remote computing, several steps must occur. First, both parties must have appropriate Intel CPUs with SGX capabilities [6]. Then the desired SGX program must be written. Out of the box this can only be done in C or C++. An SGX program is rather unique, as it is separated into two separate parts, the secure and unsecure portions. The secure portion is what is executed in the enclave, whereas the unsecure portion can handle I/O and non-sensitive operations and data [6]. These two can interact, however, to reduce attack vector size, the number of interactions and the size of the secure portion should be kept to a minimum. Once the desired program is created, an enclave on both the local and remote computer are made. These can then perform attestation, where a hash is made for both enclaves and compared to see if the desired program is truly on the other side. Enclave code cannot be modified after creation so if the attestation passes, it should be secure. The encrypted data can now be

passed to the remote enclave, which will then decrypt and perform its program on the data, then encrypt and send the results back [6].

3.2 GWAS with Intel SGX (DyPS)

Genome-Wide Association Studies (GWASs) are studies that look at genome-wide sets of genetic variations to try and ascertain whether they correlate with a given trait. In particular, they tend to look at whether single-nucleotide polymorphisms (SNPs) are associated with a given human disease. Because of the nature of the work, any one institution can rarely complete such a study themselves, as it requires extremely large volumes of data that, due to the difficult nature of collecting, processing, and storing large-scale data, few institutions could have on their own [7]. As such, it is very common for GWASs to include many organizations contributing their data to complete the study collectively. However, genetic information is extremely sensitive personal data (it is considered PHI by HIPAA) [8] that cannot be easily be de-identified (genetic data is by its nature identifying) and may be subject to different privacy regulations due to institutions being in different locales. This means that there is much interest in finding ways to do GWASs remotely between many different institutions without any of them revealing any information to each other, which is exactly the sort of problem SGX can solve [7]. This is why the examples illustrated in this paper will revolve around this use case. However, SGX would be able to be used in similar scenarios studying different healthcare topics.

The Dynamic, Private and Secure GWAS (DyPS) project was an attempt to utilize Intel SGX enclaves to create a system that would allow any number of organizations to contribute to a GWAS without revealing any of their information to other parties, without being vulnerable to some common attacks on GWASs, and remaining dynamic, able to give reports as each batch of information is input and processed and allowing the number of SNPs studied to change as more data is added [9]. DyPS works by having an enclave in a remote computer accessible to all interested parties. Each party then individually does attestation with the enclave, during which the enclave sets up a unique symmetric private key with each party that can be used to send future data. As organizations send in their data, DyPS processes them in first-in-first-out (FIFO) order. DyPS then seeks to create batches of data that do not lead to one of its

security requirements being invalidated, thus preventing many attacks and privacy leaks [9]. It then takes this batch into the enclave for GWAS processing, updating its statistics. These statistics are then compared to another set of rules that check whether a membership attack (an attacker checking whether a known genome is within a given dataset) is plausible on the aggregate statistics. If it is not, it finally releases the statistics to the researchers [9].

The result of the DyPS process is that, compared to the fastest non-dynamic (as DyPS was the first dynamic GWAS program) GWAS algorithm, DyPS had a reasonable comparative processing delay (11%) while having no privacy leaks whereas the comparison algorithm leaked 8% of the genomes [9]. It also protects against several common attacks against GWASs, such as membership and recovery (the re-identification of an individual whose genome was a part of the study) attacks. It also protects against bio-centers attempting to collude together to learn information, however, it does not protect against data poisoning, as this is an open question in this line of study as genomes can be forged. It is also vulnerable to attacks against SGX itself, which will be discussed later [9].

3.3 GWAS with SGX and Homomorphic Encryption (SAFETY)

Another, vastly different approach to GWASs was taken with Secure gwAs in Federated Environment Through a hYbrid solution with Intel SGX and Homomorphic Encryption (SAFETY) project [7]. SAFETY has two main differences with DyPS. The first is that in SAFETY all of the data is stored on the individual data contributors' systems until they are needed for processing, only then are they brought to the central server through a query from a researcher. The second is the use of homomorphic encryption for much of the encryption and processing of the data.

A short explanation of homomorphic encryption is that it is encryption done in such a way that computations may be done on the data without decrypting it [7]. SAFETY specifically uses a partial homomorphic system called the Paillier cryptosystem, the most important properties of which are probabilistic encryption and addition homomorphism. Probabilistic encryption means that each time a given plaintext is encrypted it generates a different ciphertext. Addition homomorphism means that given

ciphertexts c_1 and c_2 and any public key n, $c_1 + c_2 = (c_1 * c_2) \bmod n^2$, meaning that it is possible to do addition over ciphertexts [7].

SAFETY involves four types of entities, a researcher, data owners, a Crypto Service Provider (CSP), and an SGX-enabled central server [7]. The CSP gives the data owners a public key to encrypt their data before anything else. It then sends the private key to the enclave without the central server seeing it. When a researcher wants to perform a test, they query the central server, which then asks the individual data owners for any relevant information. This encrypted data is then sent to the central server, which performs homomorphic addition and sends it into the enclave where it can be decrypted for further computations that would be too intensive to do homomorphically. Finally, the results of these computations are sent to the original researcher [7].

The SAFETY algorithm relying on only doing computation inside of the SGX enclave faces a modest overhead of 8% [7]. However, when homomorphic encryption is implemented SAFETY is 1.7 to 4.82 times faster than SGX alone, with the comparative efficiency scaling with the number of data owners. The results shown in the two papers [7,9] are not easily comparable to each other, but they both show a fundamentally different approach to the same problem. DyPS is more centralized, with all of the data being stored on one central server, and processed in a FIFO manner [9]. SAFETY instead decentralizes everything until an actual query is made, when data is then requested from the data owners [7]. Arguably neither approach is better than the other, the centralized approach would mean that the central server would need to have a lot of storage capacity as well as the ability to quickly process the data. However, it actively processes the data rather than waiting until a request is made, as well as likely taking some data storage burden off of the data owners. The decentralized approach likely means fewer costs towards the central server, however, a trusted CSP is needed and each data owner must now keep track of its data. Also, unlike DyPS, the SAFETY paper makes no claims to protection from any given attacks and also may be vulnerable to the same attacks against SGX that DyPS may be, as the homomorphic encryption is only outside of the enclave and thus grants no further protection [7,9].

3.4 SGX Drawbacks and Problems

As with any technology, SGX has its drawbacks and problems. For one, programming in SGX can be rather complex. Having to split one's program up into safe and unsafe portions is not always easy, and the goals of minimizing the size of the safe portion and minimizing interactions between the safe and unsafe portions can often be contradictory. Many healthcare institutions may not have skilled enough programmers to create such programs. Of course, these organizations could use third-party programs, but then they have to trust whoever wrote them with their security.

Trusting third parties is a problem often encountered in both remote computing and SGX. For remote computing, even if one trusts the HaaS provider (which one should not), that does not completely alleviate the issue, as they could be compromised by malicious actors, or they could simply go out of business, meaning all of the data may be lost. SGX should theoretically protect from malicious third parties, but it opens up interactions with another third party, Intel. Intel SGX is only available on certain Intel processors, meaning that any sort of system involving SGX forces all parties involved to purchase Intel products [6]. Intel also has complete control of SGX, and as such putting trust in SGX means putting trust in Intel. Finally, SGX has the built-in ability to force any users attempting to perform attestation to go through a third party (Intel) first. This seems to be for a potential licensing scheme, meaning that if Intel SGX ever became the dominant TEE, Intel could possess the ability to choose winners and losers in the field [6].

Of course, data may not even be truly safe with SGX, as SGX has potential vulnerabilities to some attacks, most notably software side-channel attacks [6]. Valuable information can be gathered from the nature of how the enclave operates which breaks semantic security. "For example, if the number of cycles taken by an integer ALU… depends on its inputs, the snooping LP could learn some information about the numbers multiplied or divided by the other LP" [6].

4 Suggestions

With the security concerns raised in section 3.4, it is the opinion of this author that SGX alone is not enough to consider remote computing completely secure for healthcare. However, it, or similar TEEs,

are still likely the best solution for secure remote computing now and in the future, an opinion shared by Confidential Computing Consortium (CCC) [10].

Further technical research would reside in one of two fields. The first is finding more applications for SGX. Currently, most of the research about SGX in the healthcare field seems to revolve around GWASs. This does make sense, as their nature seems to make a perfect case study. However, more research should be done into using SGX for other applications. In particular, GWASs involve multiple parties all performing research together, and as such, more research should be done into using SGX for single parties. The other field that SGX research would be in is how to make the technology more secure, whether that be trying to find more problems in it or more solutions for those problems. The side-channel attacks in particular need to be addressed before SGX can find mainstream use.

Of course, no technical research can fix the ethical problems associated with Intel's control of SGX. This is unfortunately a problem with few solutions. An alternative can be sought, however, TEEs are often quite intertwined with the hardware of the machine, and any that are not would have to find solutions to a whole new host of security issues. The best solution would likely be for Intel to be more open with SGX, along with dropping any further licensing ideas, however, this seems unlikely, as Intel would greatly benefit from its position if SGX does become mainstream.

5 Conclusions

This paper looked at the potential use of Intel SGX in the healthcare industry for the purposes of remote computing. Intel SGX is a set of instructions that allows for the creation of encrypted and protected portions of memory called enclaves, that should be protected from any physical or software attacks, including from software with higher privilege levels. It also allows for attestation to send encrypted information to the enclave with no other parties being able to see the plaintext. This would allow healthcare organizations to perform costly computations using a third-party's hardware, limiting their expenses while also protecting the information for purposes of HIPAA and similar laws. Several theoretical schemes for this have been proposed, especially for GWASs, as seen with DyPS and SAFETY.

As it stands, SGX as it works on a theoretical technical level would be close to a perfect solution for secure remote computing, not just in the healthcare industry, but everywhere. In reality, there are some potential security issues with SGX that may dissuade companies from utilizing it currently. These issues may be fixed, but against an attacker that knows what they are doing, they currently could be devastating. In addition, the ethical and logistical issues surrounding SGX could further dissuade healthcare companies from utilizing it, as they may not want to be so closely bound to Intel to securely process their data. SGX and similar TEEs will likely be the answer for healthcare companies in need of remote computing in the future, but it may not be an answer now. If the security issues the technology faces are fixed or minimized and the ethical questions surrounding it are answered, then it will do a tremendous good for the healthcare industry.

6 References

[1] Journal, H. (2022, February 12). Why is HIPAA Important? Updated 2022. *HIPAA Journal*. https://www.hipaajournal.com/why-is-hipaa-important/

[2] HIPAA History. (n.d.). *HIPAA Journal*. Retrieved November 28, 2022, from https://www.hipaajournal.com/hipaa-history/

[3] Rights (OCR), O. for C. (2008, May 7). *Summary of the HIPAA Privacy Rule* [Text]. HHS.Gov. https://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html

[4] Rights (OCR), O. for C. (2009, November 20). *Summary of the HIPAA Security Rule* [Text]. HHS.Gov. https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html

[5] Confidential Computing Consortium. (2021, January). *Confidential Computing: Hardware-Based Trusted Execution for Applications and Data*. https://confidentialcomputing.io/wp-content/uploads/sites/85/2022/11/CCC_outreach_whitepaper_updated_2022-11-02.pdf

[6] Costan, V., & Devadas, S. (2016). *Intel SGX Explained*. https://eprint.iacr.org/2016/086

[7] Sadat, M. N., Aziz, M. M. A., Mohammed, N., Chen, F., Wang, S., & Jiang, X. (2017). *SAFETY: Secure gwAs in Federated Environment Through a hYbrid solution with Intel SGX and*

Homomorphic Encryption (arXiv:1703.02577). arXiv. https://doi.org/10.48550/arXiv.1703.02577

[8] Rights (OCR), O. for C. (2007, March 28). *Does the HIPAA Privacy Rule protect genetic information?* [Text]. HHS.Gov. https://www.hhs.gov/hipaa/for-professionals/faq/354/does-hipaa-protect-genetic-information/index.html

[9] Pascoal, T., Decouchant, J., Boutet, A., & Esteves-Verissimo, P. (2021). DyPS: Dynamic, Private and Secure GWAS. *Proceedings on Privacy Enhancing Technologies*. https://doi.org/10.2478/popets-2021-0025

[10] *Confidential Computing Consortium—Open Source Community*. (n.d.). Confidential Computing Consortium. Retrieved November 30, 2022, from https://confidentialcomputing.io/