VoiceGuard: Secure and Private Speech Processing

Ferdinand Brasser[1], Tommaso Frassetto[1], Korbinian Riedhammer[2], Ahmad-Reza Sadeghi[1], Thomas Schneider[1], Christian Weinert[1]

[1]Technische Universität Darmstadt, Germany
[2]University of Applied Sciences Rosenheim, Germany

{ferdinand.brasser, tommaso.frassetto, ahmad.sadeghi}@trust.tu-darmstadt.de, korbinian@ieee.org, thomas.schneider@cs.tu-darmstadt.de, christian.weinert@crisp-da.de

Abstract

With the advent of smart-home devices providing voice-based interfaces, such as Amazon Alexa or Apple Siri, voice data is constantly transferred to cloud services for automated speech recognition or speaker verification.

While this development enables intriguing new applications, it also poses significant risks: Voice data is highly sensitive since it contains biometric information of the speaker as well as the spoken words. This data may be abused if not protected properly, thus the security and privacy of billions of end-users is at stake.

We tackle this challenge by proposing an architecture, dubbed VoiceGuard, that efficiently protects the speech processing task inside a trusted execution environment (TEE). Our solution preserves the privacy of users while at the same time it does not require the service provider to reveal model parameters. Our architecture can be extended to enable user-specific models, such as feature transformations (including fMLLR), i-vectors, or model transformations (e.g., custom output layers). It also generalizes to secure on-premise solutions, allowing vendors to securely ship their models to customers.

We provide a proof-of-concept implementation and evaluate it on the Resource Management and WSJ speech recognition tasks isolated with Intel SGX, a widely available TEE implementation, demonstrating even real time processing capabilities.

Index Terms: speech recognition, privacy protection, cloud computing

1. Introduction

Devices providing voice-based interfaces are omnipresent in today's world. Amazon Alexa, Apple Siri, Google Assistant, or Microsoft Cortana are available to the more than two billion smartphone users in 2018. Also, there is a steadily increasing number of smart-home devices, like Amazon Echo, Apple HomePod, or Google Home, solely relying on voice-based interaction. Possible application scenarios are not restricted to the consumer market but increasingly cover professional activities, for example enterprise-ready smart assistants guiding through complicated business processes in order to increase productivity.

In any of the aforementioned cases, voice data is constantly transferred to the cloud for remote speech processing, such as automated speech recognition (ASR) or speaker verification. This poses significant security and privacy risks since voice data contains sensitive biometric information as well as the spoken words: in case unprotected voice data gets out of hand, it may be abused, e.g., for impersonation attacks, assembling fake recordings, or simply extracting intimate as well as secret and sensitive content.

A naive solution to these problems is to ship the speech processing code together with corresponding models to the users to run locally. While this might be infeasible for low-end devices anyhow, it also contradicts the business interests of vendors providing such models which represent their intellectual property.

Attempts based on purely cryptographic solutions, i.e., homomorphic encryption (HE) or secure multi-party computation (SMPC), guarantee that neither user nor vendor need to reveal their respective inputs in the clear. However, as we elaborate in our review of related work in §2, these solutions are highly impractical due to their massive overhead in computation time and communication costs. Besides, none of the existing solutions considered user-specific models, i.e., the common practice to train or adapt a separate model for each user that covers deviations from the model to incorporate specific characteristics, e.g., in dialect and pronunciation.

Goals and Contributions. To overcome these limitations, we propose VoiceGuard in §5, an architecture that efficiently protects speech processing tasks using a trusted execution environment (TEE). It allows the secure processing of confidential data even in a hostile environment by combining cryptographic techniques with hardware-enforced code and data isolation.

Although the concept of TEEs has been known for many years, they only recently became widely available with Intel's introduction of Software Guard Extensions (SGX). SGX is Intel's implementation of a TEE available in most of their recent CPUs. It generated large interest in both academic research and industry: Signal, for example, a popular instant messaging service similar to WhatsApp, employs Intel SGX to identify the contacts in a new user's address book that are signed up to the service while all other contacts remain private [1]. The deployment of such privacy-preserving services is also facilitated by leading cloud service providers (e.g., Microsoft Azure [2]) making this CPU feature available to customers.

VoiceGuard enables secure and private speech processing, independent of who actually controls the machine performing the computation. Thus, it could be hosted by the vendor of the speech processing software, a third party service provider, or even the user. The latter on-premise solution could be preferred if it is necessary to comply to certain legal regulations or the user wants to exclude the possibility of a malicious party performing sophisticated hardware attacks.

The architecture of VoiceGuard can easily be extended to enable user-specific models, such as feature transformations (including fMLLR), i-vectors, or model transformations (e.g., custom output layers). We present a fully functional prototype implementation of VoiceGuard for ASR based on the kaldi toolkit [3]. Moreover, we conduct an empirical performance evaluation of the Resource Management and WSJ speech recognition tasks in §6, thereby demonstrating that the overhead induced by our protection measures is low enough to enable privacy-preserving speech recognition in real time.

2. Related Work

In the following, we briefly review general approaches for privacy-preserving machine learning (grouped by the underlying technology) that could be adapted to speech processing tasks which depend on the evaluation of neural networks. Furthermore, we review specialized approaches for various privacy-preserving speech processing tasks.

2.1. Privacy-Preserving Machine Learning

Secure Multi-Party Computation (SMPC). SMPC enables two or more parties to jointly compute a publicly known function without revealing private inputs to each other by executing an interactive cryptographic protocol. Recently, SMPC protocols and frameworks have been applied to both privacy-preserving training of neural networks [4] and corresponding inference [5, 6, 7, 8, 9], mostly for image classification tasks. However, compared to unprotected data processing, SMPC-based solutions require several orders of magnitude higher computation time and communication cost. They are especially impractical for on-the-fly processing due to repeated initialization costs.

Homomorphic Encryption (HE). HE allows performing operations on encrypted data s.t. the decryption of the computation result equals the outcome when performing the same operations on plaintext data. Microsoft CryptoNets [10] was the first attempt to utilize HE for secure evaluation of neural networks, followed by an improvement named CryptoDL [11], which replaces complex activation functions with approximated low-degree polynomials. Nevertheless, the reported performance results indicate that solutions based on heavyweight HE are currently far from suitable for speech recognition in real time.

TEE. SMPC via TEEs has been proposed in [12, 13, 14]. Ohrimenko et al. [15] adapt several machine learning algorithms, including neural networks, to prevent cache-based side-channel attacks in scenarios where multiple institutions use Intel SGX to securely share their datasets for training and evaluation of joint machine learning models. In [16], the authors introduce a similar protection mechanism that is efficient enough for real-time data processing: instead of preventing memory accesses that depend on sensitive data, they add noise to memory traces by accessing dummy data. The very recent Chiron [17] system allows a user to train a model using the computing resources of a cloud service provider while the training data remains hidden and the resulting model can only be accessed as a black box. This machine learning as-a-service (MLaaS) concept differs from our scenario where we assume vendors who provide existing models which should only be evaluated obliviously.

2.2. Privacy-Preserving Speech Processing

Pathak et al. [18] explored how to use the previously mentioned SC and HE techniques for privacy-preserving versions of speech processing tasks such as speech recognition and speaker verification. However, with their prototype implementation based on the Paillier HE scheme, it takes more than 3 hours to encrypt 1 s of audio and to recognize a single word out of a 10 word vocabulary. Admitting the impracticality of this approach, the authors furthermore propose a very efficient solution based on secure string-matching. Unfortunately, this approach can only be used for certain tasks such as speaker verification.

Recently, Glackin et al. [19] proposed an architecture for finding outsourced (encrypted) speech documents that contain given keywords. The architecture works as follows: (I) the client translates audio to phonetic symbols using a CNN-based acoustic model, (II) the encrypted phones and a search index are sent to a server, and (III) the server uses a searchable encryption scheme to deliver outsourced data matching the given keywords. However, this approach requires the vendor to hand the acoustic model to the user in the clear.

3. Background

For the remainder of the paper, we assume familiarity with state-of-the-art speech processing pipelines and restrict the background to the introduction of Intel SGX.

Intel SGX. Intel Software Guard Extensions (SGX) enables processing of confidential data on untrusted systems [20, 21, 22, 23]. SGX introduces the concept of enclaves, which are programs executed in isolation from all other software on a system, including privileged software, like the operating system (OS) or a hypervisor.

Enclaves are loaded as part of a host process and are embedded in its virtual memory, like a library. The initial content of an enclave is loaded from unprotected memory, hence, it can be manipulated and is not kept confidential. Therefore, confidential data must be provisioned to an enclave over a secure channel after it has been created. However, to ensure that secret data is not sent to a malicious (or maliciously modified) enclave, the integrity and authenticity of an enclave needs to be verified before provisioning secret data. To enable this, SGX provides a security service called remote attestation (RA). With RA, an external party can verify whether an enclave was created correctly, i.e., a cryptographic hash of the initial memory state of an enclave is signed by the platform signing key which is built into the CPU.

Once available inside an enclave, secret data can be encrypted using an enclave-specific key and written to untrusted storage, e.g., the hard disk. This sealing mechanism allows an enclave to use secret data across multiple instantiations.

4. Model and Assumptions

In this paper we consider a setting where three parties collaborate to perform secure and private speech processing:

(1) The user provides the voice data to be processed. She is concerned about her privacy and does not want the other parties to identify her based on biometric characteristics in her input. Additionally, the user does not want to reveal the content of her input to the other parties, i.e., they should not be able to access the voice data or the processing results. Lastly, the user does not want to be traceable across multiple sessions.

(2) The vendor provides the software required for speech processing together with corresponding models. This data constitutes the vendor's intellectual property, hence it must be kept confidential from the other parties.

(3) The service provider carries out the actual computations based on the user's and the vendor's inputs. The service provider could be an independent third party, e.g., a cloud service provider. Without loss of generality, the service provider could also be under the control of the user or the vendor.

Adversary Model. The adversary's goal is to extract sensitive information, i.e., the intellectual property of the vendor, the input of the user, or data that allows the adversary to identify or track the user.

We assume that the adversary is in control of the service provider's infrastructure, in particular, all computer systems involved in performing the speech processing task. The adversary has full control over the software in the service provider's infrastructure, including privileged software like the OS or a