SecHeadset: A Practical Privacy Protection System for Real-time Voice Communication

Peng Huang[1], Kun Pan[1], Qinglong Wang[1,2], Peng Cheng[1,2], Li Lu[1,2], Zhongjie Ba*[,1,2], Kui Ren[1,2]
[1] The State Key Laboratory of Blockchain and Data Security, Zhejiang University
[2] Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
Hangzhou, Zhejiang, China
{penghuang,pankun,qinglong.wang,peng_cheng,li.lu,zhongjieba,kuiren}@zju.edu.cn

Abstract

Voice communication is convenient while also poses risks of privacy leakage, due to potential interception or eavesdropping during voice transmission. Current protections of voice privacy are almost entirely controlled by communication service providers (CSPs), which operate as a black-box to users thus hard to fully trust. To take back the control of user privacy, in this paper, we introduce `SecHeadset`, an end-to-end solution for secure voice communication based on voice obfuscation, which is plug-and-play and compatible with various CSPs. Our solution involves two parts. First, we design a voice-like noise masking scheme for voice obfuscation. The noise, mimicking voice characteristics, could effectively obscure users' voices while demonstrating resilience against noise reduction methods. Second, we develop a protocol that enables efficient channel state estimation and secure information exchange between two communication entities. Based on this information, we propose a lightweight algorithm for voice retrieval during communication. We develop a prototype of `SecHeadset` and evaluate its performance with 8 widely-used applications, including Telegram and Skype. It reduces the voice recognition accuracy of various adversaries to below 15% while maintaining communication quality. We also integrate `SecHeadset` with off-the-shelf portable devices and verify its real-world effectiveness.

CCS Concepts

• **Security and privacy** → *Privacy-preserving protocols.*

Keywords

Secure Voice Communication, Voice Obfuscation.

ACM Reference Format:
Peng Huang[1], Kun Pan[1], Qinglong Wang[1,2], Peng Cheng[1,2], Li Lu[1,2], Zhongjie Ba*[,1,2], Kui Ren[1,2]. 2025. SecHeadset: A Practical Privacy Protection System for Real-time Voice Communication. In *The 23rd Annual International Conference on Mobile Systems, Applications and Services (MobiSys '25), June 23–27, 2025, Anaheim, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3711875.3729142

* Coressponding author

MobiSys '25, Anaheim, CA, USA

ACM ISBN 979-8-4007-1453-5/2025/06
https://doi.org/10.1145/3711875.3729142

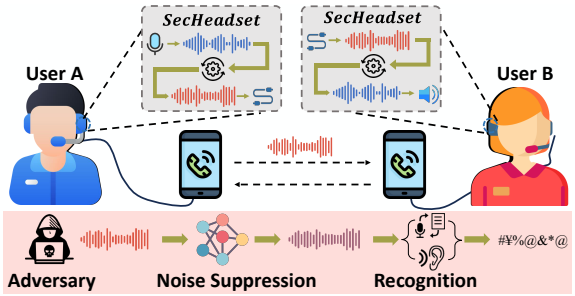

Figure 1: `SecHeadset` prevents adversaries from eavesdropping on speech contents in voice communication.

1 Introduction

Voice communications have seen rapid growths [1], which raise privacy concerns. Transmissions of voices pose risks of unauthorized access [2–4]. These voice data could be exploited for illegal surveillance, identity theft, and exposure of confidential information. To prevent this, communication service providers (CSPs) have implemented many privacy-preserving techniques, which typically operate as black-box to users thus require full trust. However, CSPs are not always reliable and may secretly collect user data [5]. There also could be flawed implementations in encryption algorithms [6, 7] and software vulnerabilities [8, 9], which may enable adversaries extract voice data from encrypted data stream. Besides, government surveillance programs [10] may compel CSPs to provide access to user voice communications

These limitations have prompted the development of white-box user-controlled privacy-preserving techniques without the reliance on CSPs [11–16]. These methods encrypt the voice before feeding it to CSPs and decrypt it after receiving. However, unlike CSPs that fully control the voice transmission channel, users have limited control over that. This channel, inherently lossy from an end-to-end perspective, could corrupt the ciphertext thus making these methods unfeasible.

Based on these dilemmas, we summarize several capabilities that a privacy-preserving system for voice communication should have for practical usage: 1) the system should be user-controlled and provide end-to-end protection without requiring the cooperation of CSPs. Except for the communication participants, other entities such as CSPs and communication devices should be prevented from accessing privacy from the transmitted audio. 2) the system should not significantly affect the communication quality while providing a seamless user experience. Specifically, it should be plug-and-play, requiring little effort to set up and operate. 3) given the variety of CSPs that users might employ, the system should be adaptable across different CSPs.

In this work, we propose a privacy-preserving system for voice communication leveraging the idea of voice obfuscation. Compared to encryption, voice obfuscation is robust to distortions induced by lossy channels, making it feasible in practical usage. As shown in Fig. 1, our system contains two parts: voice obfuscation and retrieval. On the transmitter, we introduce carefully designed noises to obfuscate voices before transmitting to smart devices, On the receiver, the same noises are generated and removed from obfuscated voices before playback to users. The system works as middleware that relays audio between users and smart devices.

To achieve the aforementioned capabilities, we introduce several novel techniques. First, to guarantee end-to-end security, the noise we use should be robust to denoising techniques, as adversaries are likely to utilize various denoising techniques to remove our noise after obtaining the transmitted voice. To be resilient to these denoising techniques, in this paper, we design a new type of phoneme-based denoising-resilient noise. Because of its speech-like characteristics, it can effectively obfuscate voices while exhibiting strong resistance against various denoising methods. Experiments show that our noise maintains its robustness even against targeted-trained denoising networks.

Second, to provide a seamless user experience and maintain communication quality, the system should be able to retrieve original voices from the obfuscated voice on the receiver side. To achieve this, we design a protocol for real-time information exchange between communication participants, which enables the transmitter and the receiver to share knowledge required for voice obfuscation and retrieval. Based on these, we develop an adaptive, lightweight, and spectral-based voice retrieval algorithm for retrieving the original voice automatically. This algorithm is computationally efficient, making it possible to be implemented on resource-constrained portable devices for real-time voice communication.

Third, as the distortion induced by each CSP is different, to ensure adaptability, we investigate influential factors that contribute to channel distortions and propose methods for measuring them in real-time. These factors are adopted to facilitate the voice retrieval algorithm to make it adaptive to different CSPs without extra configurations. Experimental results show that this method makes the voice retrieval algorithm effective in most of the tested CSPs.

Based on these techniques, we implement `SecHeadset`, which can safeguard user privacy during voice communications. The system is plug-and-play and can adapt to various applications without prior configurations. We further integrated it into off-the-shelf portable devices and verified its effectiveness in real-time scenarios*. In general, we summarize our contributions as follows:

- We introduce `SecHeadset` for privacy-preserving voice communications based on the idea of voice obfuscation. To realize it, we design a new type of phoneme-based denoising-resilient noise for voice obfuscation, and an adaptive, lightweight, and effective algorithm for real-time voice retrieval.
- We propose a simple yet effective protocol for real-time information exchange and channel distortion estimation, allowing the system to adapt to different scenarios without prior configurations.
- We evaluate our system across eight widely-used applications. Results show that it reduces the voice recognition accuracy of adversaries to below 15% while preserving the communication quality. We also integrate our system into off-the-shelf portable devices and verify its effectiveness in real-time communications.

*The code and a demo could be found in https://github.com/desperado1999/SecHeadset

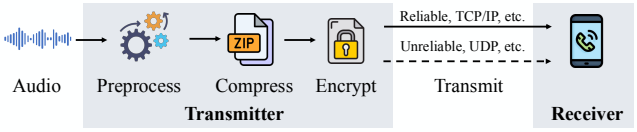

Figure 2: Audio processing in voice communication.

2 Preliminary

Voice Communication: Voice communication can be roughly classified into telephony, voice message, and VoIP. Figure 2 shows the typical audio processing pipeline for them. During communication, the transmitter would preprocess, compress, encrypt, and transmit the audio data to the receiver.

Preprocessing: The audio is first processed by voice activity detection, echo cancellation, and noise suppression. Audio components except for voices would be suppressed. Noises are commonly identified based on spectrum, as voices are always non-stationary and consist of harmonics and resonances, while noises are often stationary with a flat spectrum.

Compression: To reduce the transmission overhead, the audio is lossy compressed before transmission. To maintain the perceived audio quality, most of these compression algorithms use psychoacoustics to locate components that less affect human perception, such as the masking effect and the absolute threshold of hearing. According to network conditions, CSPs choose different compression ratios. For example, Telegram adopts the opus codec [17] and supports audio bitrates from 6 kbit/s to 510 kbit/s.

Encryption & Transmission: Before transmission, the audio data is encrypted in packet-wise. The transmission could be either reliable or unreliable. Here reliable means complete delivery without packet loss. Compared to UDP, TCP introduces less distortions to the audio data. Since the encryption is conducted packet-wise, the packet loss would not affect the decryption of other packets.

Automatic Speech Recognition (ASR) systems convert audios into text, typically containing three components: acoustic model, pronunciation model, and language model. Acoustic model converts audio features into minimum recognition units, such as phonemes. Pronunciation model decodes phoneme series into words. And language model selects the most possible word sequence. The correct recognition of the minimum units is crucial for correct results.

Voice Enhancement and Separation are common techniques for audio processing, both of which could be used for denoising. Enhancement processes one single input to produce one cleaner output, while separation divides one audio stream into multiple sources. These methods leverage differences among sources for denoising. When sources share similar attributes, such as timbre, the denoising process would be challenging.

3 Problem Formulation & Related Work

3.1 System Model

In this work, we aim to build a privacy-preserving system for voice communications, which is fully transparent and user-controlled. As in Fig. 3, our system model contains three types of entities:

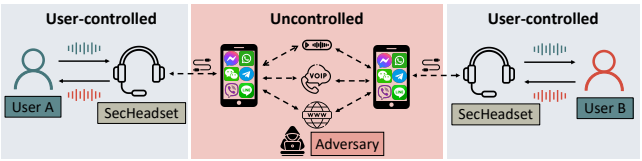

Figure 3: System Model.

`Users`: We assume two mutually-identified `users` are engaging in voice communications over smart devices. To protect their voice privacy, both of them would first register the `SecHeadset` and then use it during communications.

`SecHeadset`: `SecHeadset` works as an audio relay between users and smart devices. During communications, `SecHeadset` records voices from the user, obfuscates it to prevent recognition by potential `adversaries`, and transmits it to the user's device. When receiving audios from the device, `SecHeadset` retrieves voices and then plays them back to the user. From the user's perspective, `SecHeadset` works as a normal headset, requiring no modifications to the communication process or assistance from CSPs.

`Adversary`: The `adversary` aims to eavesdrop on the communication content. It could be the communication software, CSPs, or any others capable of accessing the data transmitted between smart devices. Please note that we assume the hardware and operating system of the communication device are trustworthy, since malicious hardware or operating systems are able to use internal microphones for eavesdropping even when external audio devices are in use, which is beyond this paper's scope.

3.2 Threat Model

Adversary's Goal: The adversary aims to eavesdrop on the content of users' voice communication.

Adversary's Knowledge: We assume the adversary knows the detail of `SecHeadset`, including the stored original data and signal processing algorithms. Compared to legitimate users, the adversary only lacks access to user-specific and communication session-specific information stored in `SecHeadset`. The former is generated during the registration process and the latter is randomly generated for each communication session.

Adversary's Capabilities: We consider an adversary who can acquire the voice transmitted between smart devices during communications. Please note that this transmitted voice has been obfuscated by `SecHeadset`. The adversary could acquire the voice data either before transmission (not affected by channel distortions) or after transmission (affected by channel distortions). Such an adversary could be the CSP itself, which has legitimate access to the transmitted data, or a third-party adversary who exploits security vulnerabilities (such as [8, 9]) to extract voice information from the encrypted data stream.

Based on these capabilities, to eavesdrop on the communication content, the adversary will:

• Use various voice data to query the `SecHeadset` to get sufficient input and output data pairs, then use these data pairs to train a specialized audio denoising model.

• Upon obtaining the voice data transmitted during the communication, employ the specialized denoising model to enhance the voice data obfuscated by `SecHeadset`.

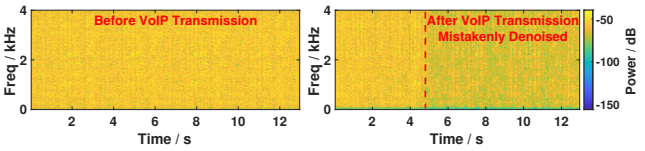

Figure 4: Encrypted voice data being mistakenly denoised.

• Subsequently, use advanced ASR systems to extract speech content from the enhanced voice signal.

3.3 Design Goals

Security: The system should prevent the leakage of voice content during communications. Other information such as user identities are not considered.

Usability: The system should not significantly affect communication quality and should not require many user operations. It could be used as a normal headset, being plug-and-play and requiring no additional configuration for different devices and CSPs.

Flexibility: The system should be adaptive to various scenarios, including different communication types and different CSPs.

3.4 Related Work

Many works focus on providing user-controlled privacy during voice communications, with similar system models as shown in Fig. 3. Most of these works adopt encryption, either by directly encrypting audio features or by embedding the audio data into cover audio based on encryption algorithms.

Direct encryption: [11] proposes a chaos-based method, which encrypts audio blocks in the time domain to ensure secure audio data transmission over insecure networks. [12] proposes a novel approach to encrypt audio data and implements it on FPGA to ensure real-time processing. *Encryption with cover audio*: [13] proposes an audio steganography algorithm based on discrete wavelet transform (DWT) and singular value decomposition (SVD) for convert speech communication. [18] converts audio waves into sign bits and amplitude bits and then hides them into a cover audio. In [15], the target speech signal is compressed using discrete cosine transform (DCT). The cover audio is decomposed with DWT and SVD, then the DCT coefficients are embedded into the singular matrix of the cover audio with a chaotic map.

However, most of these methods are designed based on ideal transmission channels and have not considered distortions induced in real-world voice communication, making them unfeasible for practical usage. In [16], the authors introduce a novel distortion-tolerated encryption scheme, which is claimed to be robust to compression algorithms. While through experiments we find that it still suffers from noise suppression processes that are common in most communication applications, which makes this method still impractical. We will show more details in the following part.

4 System Overview

4.1 Why Encryption-based Methods Failed?

As mentioned earlier, encryption-based methods for secure voice communication could be classified into CSP-controlled and user-controlled. The former is not transparent to users and there could be flawed encryption protocol implementations [6, 7], government surveillance programs [10], and dishonest CSPs that secretly collect

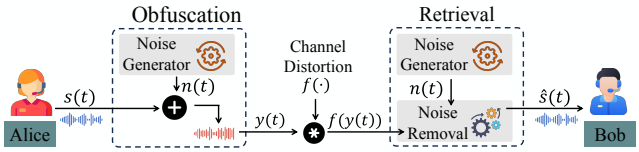

Figure 5: Voice flow in the proposed system.

user data [5]. For the latter, as shown in Fig. 2, unlike CSP-controlled methods that can manage audio processing procedures within transmitters and receivers, user-controlled methods can only manipulate audio before inputting into transmitters and after outputting from receivers, which limits user-controlled methods to encrypt audio out of transmitters. However, the preprocessing and lossy compression within transmitters would disrupt the encrypted data, making it hard to decrypt. Moreover, as the encrypted data typically exhibits flat spectrums, it could be identified as environmental noises by communication applications, leading to incorrect suppression and further rendering decryption unfeasible. Take encrypted audios provided by [16] as an example, as shown in Fig. 4, after transmission, the encrypted audio is significantly distorted.

4.2 Proposed System

In this paper, instead of encryption, we obfuscate the voice signal by adding carefully designed noise signals. This method is advantageous because, unlike the decryption process that is sensitive to distortions, retrieving original voices from obfuscated voices is more robust. As long as the distortion remains within an acceptable range, it would only introduce extra noises in the retrieved voice rather than destroying the retrieval process.

As shown in Fig. 5, assume Alice and Bob are having a voice call over smart devices, with Alice intending to send her voice $s(t)$ to Bob. To ensure privacy, Alice first adds noise $n(t)$ to her voice and then sends the mixed signal $y(t) = s(t) + n(t)$ to Bob. The noise $n(t)$ is generated by a noise generator, which is driven by user-specific parameters and communication session-specific seeds. After traversing the transmission channel, the signal received by Bob would be $\hat{y}(t) = f(y(t))$, where $f(\cdot)$ is the induced distortion. Bob then would remove $n(t)$ from $\hat{y}(t)$ to retrieve $\hat{s}(t)$. Similar to Alice, we assume Bob has the same noise generator and knows the required parameters. While for adversaries, although they know the detail of `SecHeadset`, they cannot generate the correct $n(t)$ without access to the user-specific parameters and session-specific seeds, preventing them from extracting $s(t)$ directly. The system is supposed to be full-duplex, allowing both parties to send their voice simultaneously.

However, to achieve the goals outlined in Sec. 3.3, two main challenges must be addressed. **1) to ensure end-to-end security, the noise $n(t)$ should be robust against denosing methods.** Once the adversary obtains the obfuscated audio, they would attempt to suppress the noises with various denoising techniques. Besides, the obfuscated audio should not be identified as noises by communication applications, otherwise, it would be significantly suppressed and distorted, making the receiver hard to retrieve the original voice. Therefore, it is necessary to design a new type of noise that remains robust against these denoising techniques. And **2) to achieve usability, the receiver should be able to remove $n(t)$ from $f(y(t))$.** In the proposed system, we assume the receiver

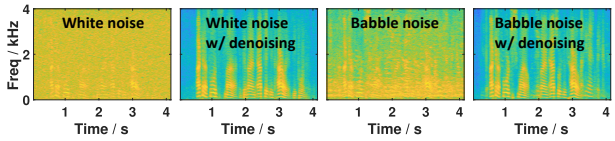

Figure 6: Noisy speech after being processed by SepFormer [20].

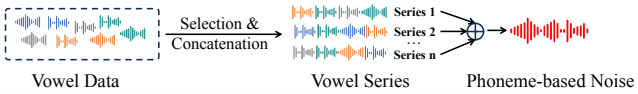

Figure 7: Generation of phoneme-based noise.

knows the detail of the noise $n(t)$ added by Alice. This requires a share of information for the noise generator, which is challenging in real-time voice communication when utilizing plug-and-play systems. Additionally, the inherent channel distortion $f(\cdot)$ makes the signal received by the receiver differ from the transmitted one, making the additive noise difficult to remove by simple subtraction. Deep learning-based methods such as [19] require offline operation thus not feasible for real-time processing. Furthermore, the transmission channel functions as a black box to the users. The distortion $f(\cdot)$ could vary across different communication applications and even the same application under different network conditions. Consequently, how to remove the noises from $\hat{y}(t)$ is challenging.

To address these challenges, we first propose a novel phoneme-based noise that is resilient to different denoising techniques while not likely to be identified as noise by existing communication applications. Next, we investigate channel distortion from multiple perspectives and propose a lightweight spectral subtraction-based voice retrieval approach that can adapt to different channel distortions. Finally, we detail the design of `SecHeadset` and the communication protocol employed for the secure exchange of information between the two users.

5 Voice Obfuscation and Retrieval

5.1 Phoneme-based Denoising-Resistant Noise

As stated in our threat model, the adversary knows the details of `SecHeadset` but lacks knowledge of user-specific and session-specific data, so generating the exact noise series as the users does is infeasible. To address this, the adversary queries `SecHeadset` to gather sufficient data and trains a specialized denoising model. In this scenraio, in addition to misleading ASR systems, the obfuscation noise should also be robust against denoising models. However, most existing noises would be easily suppressed, even for the challenging non-stationary babble noise, as shown in Fig. 6.

Inspired by the concept of informational masking [19, 21], we come up with the idea of masking critical components in voices. As discussed in Sec. 2, most ASR systems adopt phonemes or subwords (typically are combinations of phonemes) as the minimum recognition unit, distorting the phoneme series in voices could induce recognition errors. Additionally, denoising models typically rely on differences between voices and noises to facilitate noise suppression, if noises and voices share similiar attributes, the denoising process may fail. Building on these insights, we design our obfuscation noise based on continuous phoneme series, as shown in Fig. 7. For the phoneme type, we select vowels instead of consonants because 1) they take up a significantly greater share of energy in voices, making them harder to be distorted than consonants. 2)

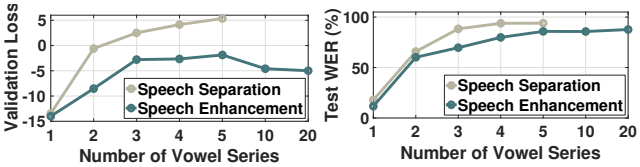

Figure 8: Validation loss and WER of denoising models for different noise structures. Higher loss and WER indicate better robustness.

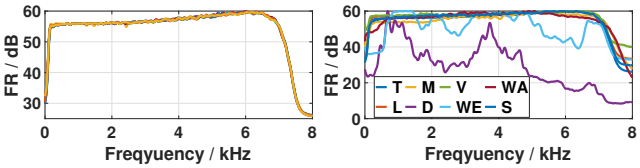

Figure 9: Normalized frequency responses (FR) of Telegram VoIP channel (left) and other applications (right).

When distorting the vowels, the adjacent consonants are likely to be also distorted due to the overlapping articulatory gestures. And 3) the energy distribution of vowels is more concentrated, making it more robust against distortions in communication channels, facilitating our voice retrieval process. For the source of vowels, we choose audios with similar timbre to the user's voice to make noises have similar attributes. For the number of vowel series, we simulate several adversaries by training specialized denoising models targeting on different series numbers to evaluate the robustness of the noise. We adopted speech separation and enhancement models based on [20]. Each result is averaged over 50 test audios.

Results in Fig. 8 show that as the number of series increases from 1 to 5, the performance of both models drops, indicating the noise has better robustness. We further evaluate 10 and 20 series against the enhancement model. Speech separation is ignored here since 5 series provides sufficient protection against the separation model, and more series would obviously make separation harder. Results reveal a slight drop in validation loss, and the test WER is nearly unchanged, indicating a slight decrease in robustness. We suspect that an excessive number of vowel series makes the noise more stationary and thus easier to remove, suggesting that both too few or too many vowel series could reduce robustness. In our design, 3 vowel series is chosen as it provides sufficient robustness while also maintaining voice intelligibility after the voice retrieval process.

5.2 Investigation of the Channel Distortion

In this part, we investigate the factors that introduce distortion during audio transmission, and then propose a spectral-based noise removal algorithm based on them. We classified these factors into network condition-related and application-related. The latter is mainly caused by the audio processing in each application, such as filtering and lossy compression.

Audio Bandwidth: We transmit 24 kHz bandwidth white noises over different channels to measure their bandwidth. We test 8 VoIP applications: *Telegram (T), Viber (V), Messenger (M),Skype (S), Line (L), DingTalk (D), WhatsAPP (WA), and WeChat (WE)* [22–29]. Tab. 1 show that these channels typically operate under 8 kHz bandwidth.

Frequency Response (FR): We transmit a chirp over these channels and measure the FR. The chirp spans from 5 Hz to 8 kHz, corresponding to the bandwidth of VoIP channels. Each measurement is repeated 10 times. The left of Fig. 9 shows FRs from 10

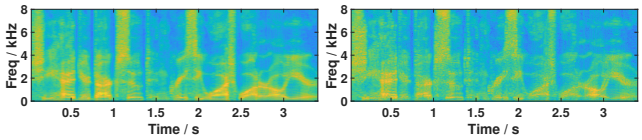

Figure 10: The original spectrogram (left) and the spectrogram of the loss (right) induced by transmission.

App	T	V*	M	S	L*	D	WA	WE
Bandwidth / kHz	8	16	8	8	16	7	8	8

∗: the audio energy drops sharply after 8 kHz.

Table 1: Audio bandwidth in each VoIP channel.

measurements of the Telegram VoIP channel, which is stable. The right of Fig. 9 shows FRs of different channels. For most applications, the FR is flat in 200 - 6500 Hz and drops outside this range. We think the reason is that applications mainly focus on human speech frequency ranges. For WeChat and DingTalk, the FR is unstable even within the human voice range. We attribute this to the use of specific denoising methods that would suppress chirp signals, which makes it hard to measure their actual FR curves.

Compression Loss: We transmit a speech signal $s(t)$ over the Telegram VoIP channel and calculate the loss in frequency domain. Assuming the received signal is $s'(t)$, the loss is defined as $||S'(f,t) - S(f,t)||$ where $S(f,t)$ and $S'(f,t)$ are spectrograms of $s(t)$ and $s'(t)$. To cover the spectrum more comprehensively, we choose a phoneme-balanced sentence from TIMIT dataset [30]: *"Don't ask me to carry an oil rag like that"*. Results in Fig. 10 illustrate a strong correlation between the loss and $s(t)$ in frequency domain. We attribute this phenomenon to the auditory masking effect [31] used in compression algorithms, which indicates a higher tolerance for loss in the presence of higher energy components.

We then aggregate the loss over time and calculate the loss ratio for each frequency. This process is repeated 10 times to ensure reliability. Results in Fig. 11 reveal consistent loss ratios in a single channel while illustrating variations between different channels. Furthermore, all loss ratio curves exhibit similar patterns: (1). High loss in frequencies out of [0.2, 7] kHz. We attribute this to the poor FR in these parts, as in Fig. 9. (2). Gradual rise of loss in [0.2, 7] Hz. We think it is caused by the error-shaping strategy employed in lossy compression algorithms. These strategies aim to preserve audio intelligibility by minimizing the loss in critical frequency bands, typically within the range of [1, 4] kHz [17]. Additionally, compression algorithms often conceal loss below the human hearing threshold [31], which is higher in high-frequency parts, indicating greater tolerance for loss.

The rest channel distortion is mainly induced by network conditions. We use network link conditioner in macOS [32] to simulate different conditions and measure the distortion.

Latency: We transmit a speech signal over VoIP channels and calculate the PESQ [33] of received signals under different latencies. Results show little impact on the audio quality when the latency<1s. When the latency>1s, VoIP calls become difficult to establish, so we ignore this scenario.

Network Bandwidth: We transmit the speech signal $s(t)$ under different bandwidths and then calculate the quality of the received audio. The bandwidth is gradually decreased until the VoIP call can not be established. In addition to PESQ, we adopt the Neurogram Similarity Index Measure (NSIM) used in VisQol [34], which

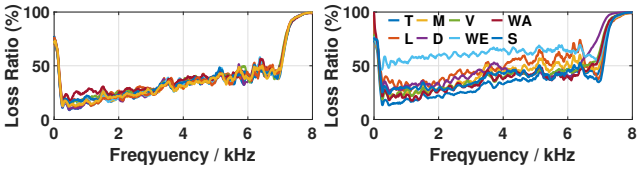

Figure 11: Loss ratio of the Telegram VoIP channel (left) and other applications (right).

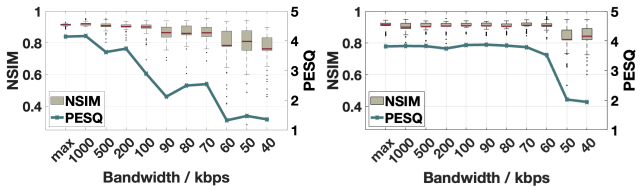

Figure 12: Impact of the network bandwidth on the Telegram VoIP channel (left) and the Skype VoIP channel (right).

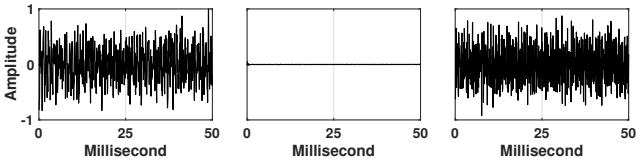

Figure 13: Left: original signal. Mid: residual after subtraction. Right: residual under a tiny misalignment.

aligns audio patch-wise thus more robust to misaligned audios. Fig. 12 shows that for the Telegram VoIP channel, PESQ gradually decreases when the bandwidth<1000 kbps, while patch-wise NSIM shows no notable variance until bandwidth decreases to 90 kbps. This suggests that low bandwidth (> 90 kbps) does not intrinsically compromise audio quality but may induce inter-patch misalignment, likely due to the compression strategy discarding silent audio pieces under limited bitrate. As bandwidth is further constrained, the mean NSIM value decreases, indicating an impact on audio quality. Skype shows similar trends, with PESQ decreasing from 60 kbps and NSIM from about 50 kbps. Both channels show an increased number of outlier patches with low NSIM at extremely low bandwidths, suggesting the presence of intra-patch misalignment.

Insights: From results in Fig. 9 - 12, we observe that all tested channels introduce significant and different distortions, which should be considered in the voice retrieval process. Specifically, the FR shows small intra-differences within each channel and greater inter-differences between channels. The transmission loss is positively correlated to the audio, and the loss ratio varies with frequency. Different bandwidths also introduce different levels of distortion. A decrease in bandwidth initially causes inter-patch packet loss. When the bandwidth falls below a certain threshold, which varies for each channel, it results in inter-patch packet loss, intra-patch packet loss, and poor audio quality.

5.3 Spectral-Based Voice Retrieval

Why time domain subtraction fails? As detailed in Sec. 4, Alice adds noise $n(t)$ to speech $s(t)$ before transmitting, and Bob removes noise components to retrieve $s(t)$ upon receiving $f(n(t)+s(t))$. An intuitive approach is to subtract $n(t)$ from $f(n(t)+s(t))$ in time domain. To test this method, we synthesize an audio signal comprising 10 different cosine waves and compress it using the libopus codec with 192kbps bitrate to simulate the channel distortion. We

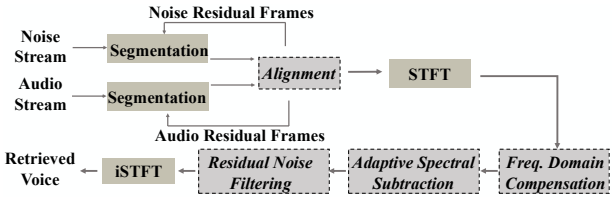

Figure 14: Overview of our voice retrieval algorithm.

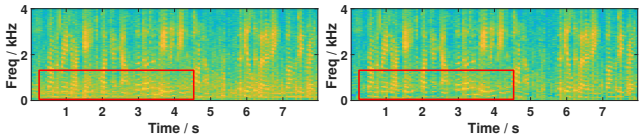

Figure 15: The estimated audio spectorgram w/o (left) and w/ (right) residual noise filtering.

then subtract the compressed signal from the original, yielding a residual. Results in Fig. 13 show that the audio can be removed effectively. However, when we shift the compressed signal by just one sample point, the residual amplifies significantly, which indicates denoising via time domain subtraction necessitates precise alignment, which is hard to meet in voice communication due to the preprocessing and channel distortion.

To bypass this precise alignment requirement, we choose to remove the noise in frequency domain, as shown in Fig. 14, which consists of segmentation, alignment, compensation, adaptive spectral subtraction, and residual noise filtering.

Prior knowledge: We assume some prior knowledge of the channel distortion, including the frequency response $FR(f)$, the compression loss ratio $LR(f)$, and the audio quality reflecting the network bandwidth. Having access to this knowledge does not constrain the generalizability of our algorithm. This knowledge would be obtained automatically at the start of each VoIP call with a probe signal. Further design details will be presented in Sec. 6.2.

Segmentation & Alignment. The first step is to align the audio with the noise signal. As audios in VoIP scenarios are stream data, we segment audios into frames and then align each frame with cross-correlation individually. The choice of the frame length is critical. A shorter length enables more frequent alignment and better accuracy. While a longer frame length reduces the computational cost. Here we choose the frame length based on the network bandwidth condition, indicated by the PESQ and NSIM scores of the probe speech. When the bandwidth is sufficient, a larger length is preferred to minimize overhead. Conversely, when bandwidth is limited, a smaller length is chosen for more precise alignment. In extremely low-bandwidth scenarios (i.e., the NSIM score is low and has large variation), due to significant packet loss, the alignment process becomes challenging and our system may be not feasible. However, in such scenarios, the original VoIP call is already unstable and the quality would be poor, thus, the privacy-preserving technique is less meaningful.

Frequency Domain Compensation. After alignment, the audio and noise frames are normalized to the same power and transformed into spectrograms with short time fourier transform (STFT). The noise is then compensated with the estimated $FR(f)$. When the $FR(f)$ is not flat, such as in WeChat and DingTalk VoIP channels, this process is skipped.

Adaptive Spectral Subtraction. After compensation, we obtain the spectrogram of the compensated audio frame $\widehat{Y}(f,t)$ and the corresponding noise spectrogram $N(f,t)$. To retrieve the estimated clear audio spectrogram $\widehat{S}(f,t)$, we adopt a multi-band spectral subtraction:

$$\widehat{S}(f,t) = \hat{Y}(f,t) \cdot (1 - \frac{||\alpha(f) * N(f,t)||}{||\widehat{Y}(f,t)||}) \tag{1}$$

where $\alpha(f)$ are unknown factors and have independent values in each frequency and could vary across frames. We use a traversal-based method to find the optimal $\alpha(f)$. Since the groundtruth $S(f,t)$ is unavailable, the optimization target is to minimize the energy of the estimated spectrogram:

$$\min_{\alpha(f)} \mathcal{L} = \sum_{f} \sum_{t} ||\widehat{S}(f,t)|| \tag{2}$$

The effectiveness of this approach lies in the fact that the noise has much larger energy than the voice. So excessively large or small values of $\alpha(f)$ will result in noise residuals in $\widehat{S}(f,t)$, increasing its energy. The reason for not using the similarity between $N(f,t)$ and $\widehat{S}(f,t)$ as the optimization target is that the spectrograms are complex values, making similarity calculation infeasible. Besides, the similarity between the power spectra (i.e., $||N(f,t)||^2$ and $||\widehat{S}(f,t)||^2$) is less distinct and not suitable for optimization.

During the traversal process, α for each frequency is first traversed from 0.2 to 1.2 in 0.1 increments. The process is then repeated around the optimal value with a finer step size of 0.01. With the optimal $\alpha(f)$, the estimated clear audio spectrogram is calculated as $\widehat{S}(f,t) = \phi(f) * \widehat{Y}(f,t)$, where

$$\phi(f_i) = \max\left\{1 - \frac{||\alpha(f_i) * N(f_i,t)||}{||\hat{Y}(f_i,t)||}, 0\right\} \text{ for each } f_i \tag{3}$$

Residual Noise Filtering. After spectral subtraction, there are still residual noises $\widehat{S}(f,t)$, as shown in Fig. 15. We think this is caused by the compression loss in each communication application, as investigated in Sec 5.2. Specifically, if the a value in the original spectrogram $S(f,t)$ is M and the compression loss at frequency f is $LR(f)$, the compressed value of M could be $M * (1+\delta)$, where $\delta \in [-LR(f), LR(f)]$.

To eliminate this, we introduce residual noise filtering based on $LR(f)$ and $\widehat{Y}(f,t)$. The main idea is that if a value in $\widehat{S}(f,t)$ is smaller than the compression error $LR(f) * \widehat{Y}(f,t)$, it could be treated as an error and we set this value to zero. Based on Eq. 3, after residual noise filtering, the estimated audio spectrogram $\widehat{S}(f,t)$ could be represented by $\varphi(f) * \widehat{Y}(f,t)$, where

$$\varphi(f) = \begin{cases} \phi(f) & \text{if} \quad \phi(f) > LR(f) \\ 0 & \text{else} \end{cases} \tag{4}$$

Fig. 15 shows that this filtering method could eliminate residual noises effectively, especially in the low-frequency part.

6 SecHeadset Design & Workflow

Overview: The workflow of SecHeadset comprises two phases: offline registration and real-time execution, as shown in Fig. 16. In offline registration, a new user must first register SecHeadset with a certification authority (CA) to obtain certification and subsequently register himself to SecHeadset by providing a few seconds of his voice recording. These registration procedures are one-time tasks for each SecHeadset. In real-time execution, the user simply powers on and connects SecHeadset to the communication device, then proceeds with voice communication as usaual. SecHeadset automatically prepares for secure communication and notifies the user when it is ready to begin.

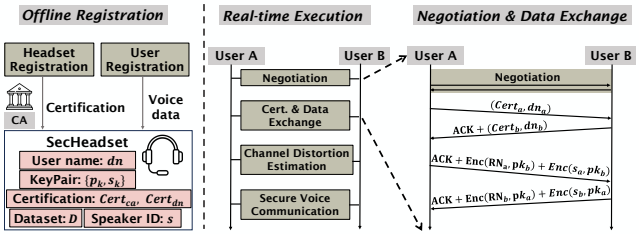

Figure 16: Workflow of SecHeadset

6.1 Offline Registration

Before registration, except for programs, SecHeadset contains only a speech dataset D, which is identical across all instances. Before usage, SecHeadset requires two one-time registration processes to obtain the necessary identification data for communication.

Headset Registration: This procedure involves generating a pair of keys, containing a public key pk and a private key sk, and submitting the user's distinguished name dn and pk to a public CA. The user would get the CA's certificate $cert_{ca}$ and his own certificate $cert_{dn}$, containing a digital signature based on his pk and dn. Headset registration relies on the public key infrastructure (PKI), and the security issues of it is not considered here. Please note that we assume all SecHeadsets are registered from the same CA.

User Registration: This procedure is used to match a speaker s who has the most similar voiceprint to the user from the speech dataset D, and the corpus of s is used for the noise generation. The user needs to record several seconds of his voice for voiceprint extraction. According to [19], a minium of 5 seconds of voice is recommended to ensure voiceprint accuracy. In this paper we assume that there is only one single user for each SecHeadset instance. User registration must be reinitiated upon any user change.

After these registration processes, in addition to D, SecHeadset contains the user distinguished name dn, the key pair (pk, sk), two certificates $cert_{ca}$ and $cert_{dn}$, the speech dataset D, and the closest speaker id s, as shown in the left of Fig. 16

6.2 Real-time Execution

Real-time Execution for Voice Call: To safeguard voice privacy during calls, users need to connect SecHeadset to the device as a normal headset, launch the call, await audio guidance, and then begin communication. Both participants in the call need to use SecHeadset. During the period between the call start and the appearance of the audio guidance, these two SecHeadsets would determine the presence of each other, negotiate a master-slave relationship for communication, exchange certificates to verify identity, and exchange essential data for the ongoing conversation. Following this, SecHeadset transmits predefined probe signals to each other for channel distortion estimation.

Real-time Execution for Voice Message: Different from the voice call, there is no real-time channel in the voice message. Therefore, we assume users will exchange their certifications and other necessary information through alternative channels before the communication, such as text messages or emails. Additionally, there is no need to estimate channel distortions since the voice message channel is stable.

6.2.1 Data Modulation. Since there are data exchange processes in real-time execution, we begin with how digital data is exchanged with VoIP channels. Specifically, we transmit digital data via frequency modulation (FM). As the audio bandwidth in most VoIP applications is limited to 8 kHz, with significant suppression out of [200, 6500] Hz (as shown in Sec. 5.2), we select 13 carriers from 800 Hz to 5600 Hz with a step size of 400 Hz. The data modulated on the first carrier is always 0, serving as a reference for data decoding. Cyclic Redundancy Check (CRC)-4-ITU [35] is adopted for error detection, so we can transmit 8-bit data in each data frame.

For synchronization during data decoding, a preamble consisting of a 3 kHz tone and two chirps ranging from 100 Hz to 7 kHz is concatenated before the modulated data. `SecHeadset` employs a two-phase method to preamble detection: a coarse detection of the single tone followed by a precise detection of the chirps using cross-correlation. This method offers reliable and efficient detection.

6.2.2 Negotiation & Data Exchange. Once powered on, the following steps are executed successively: *Negotiation.* Before communication, `SecHeadset` needs to verify the presence of its partner and establish a master-slave relationship. To achieve this, each `SecHeadset` generates a number randomly and transmits it repeatedly. Simultaneously, each `SecHeadset` listens for the partner's number. Once received, `SecHeadset` with the larger number assumes the master role. Following we assume user A is the master. *Exchange certificates.* The master (user A) sends its certificate $cert_a$ and distinguished name dn_a to the slave (user B). B then verifies A's certificate and once verified, B sends an acknowledgment (ACK) signal to inform A. Subsequently, B sends $cert_b$ and dn_b to A for verification. The preamble signal mentioned earlier is adopted as the ACK signal. *Exchange essential data.* The two `SecHeadsets` exchange essential data for the current session, including a random number RN and the speaker id s, which are used for noise generation. The data is encrypted using the partner's public key. Same as before, user A sends its data first.

6.2.3 Channel Distortion Estimation. In this part, the compression loss ratio $LR(f)$, the state of network bandwidth, and the frequency response $FR(f)$ of the channel are estimated. To get the $LR(f)$, `SecHeadset` sends a phoneme-balanced speech signal as a probe. Typically, longer probes provide better estimations, but would negatively impact user experiences due to the increased time consumption. Here we choose a 3-second probe as in Sec. 5.2, which strikes a balance between accuracy and user experience. Same as before, user A sends the probe first and repeatedly until receiving the ACK signal. Then user B repeats this process. The PESQ and the NSIM score distribution of the distorted probe are calculated to represent the state of the network bandwidth. For the $FR(f)$, it could be easily estimated with the chirp in the preamble.

6.2.4 Secure Voice Communication. When the above preparations are completed, `SecHeadset` would notify the user of the partner's name dn, indicating ready for conversation, and then start to process the audio data from microphones and connected devices. A simple illustration is in Fig. 5. Please note that `SecHeadset` is a full duplex.
• Obfuscation of voice from microphones: Before transmitting the user's voice, `SecHeadset` adds the phoneme-based noise to the voice. The noise is generated by a noise generator, which takes the dataset D, the speaker id s_a, and the random number RN_a as inputs. s_a is used for choosing the appropriate corpus and RN_a is used to set the data sampling order. The default voice SNR is -9.
• Retrieval of voice from communication devices: Upon receiving the audio transmitted by the partner, `SecHeadset` retrieves the original voice using the spectral-based voice retrieval method presented in Sec. 5.3. The noise adopted in the denoising method is also generated by a noise generator, which takes s_b and RN_b of the partner as inputs. To reduce the time delay, all these operations are performed frame-wise. As long as the time consumed for each frame is smaller than the frame length, real-time processing can be achieved. The default frame length is set to 64 ms, and the time delay introduced by `SecHeadset` would be 64 ms + frame-wise operation time.

7 Evaluation

7.1 Experimental Settings

Datasets and Tools: We utilize LibriSpeech [36] excluding the test-clean subset as the dataset D in each `SecHeadset`, containing about 1000 hours of speech. The vowel database for obfuscation noise generation is extracted from D. We choose 40 test audios from the test-clean subset. To evaluate `SecHeadset`'s performance more comprehensively, we choose audios from different speakers and prefer relatively long audios. The chosen test set contains about 2,000 words in total. When simulating an attacker that could train specialized models, we employ the train-clean-100 subset as the training set and the corresponding phoneme-based noise is generated from other subsets. RIRs Noises dataset [37] is used when testing the performance under air-traveling distortions. Network link conditioner [32] is adopted to simulate different network conditions. We use Amazon Transcribe [38] for speech recognition.
Applications and settings: We choose eight widely-used communication applications: Line (version: 14.7.2) [26], Telegram (version: 10.13.0) [22], Viber (version: 22.7.0.0 g) [23], Facebook Messenger (version: 460.0.0.48.109) [24], Skype (version: 8.120.0.207) [25], DingTalk (version: 7.5.30) [27], WhatsApp (version: 2.24.10.85) [28], and WeChat (version: 8.0.49) [29]. Without specific mention, the Telegram VoIP channel is used for evaluating the impact of various factors. All application settings are restored to defaults. The camera is disabled if available in the VoIP scenario.
Metrics: We use Word Error Rate (WER) and Short-Time Objective Intelligibility (STOI) [39] for assessment. WER is defined as $\frac{S+D+I}{N}$, where S, D, I, and N are the counts of substitutions, deletions, insertions, and total number of words in the reference, respectively. STOI estimates the perceived intelligibility using time-frequency measurement which is well-suited for noisy speech. STOI ranges from 0 to 1, with higher values indicating better intelligibility. When assessing the security of our system against adversaries, the audio obtained by adversaries should have a high WER and a low STOI. When considering the usability, the audio played back to users should have a low WER and a high STOI.
Hardware & Configuration: We launch communication between two phones (Redmi K30 Ultra and K30 Pro), with audio transferred between phones and a desktop using USB soundcards [40]. Except for the portable implementation, audios are processed by a `SecHeadset` prototype on the desktop. The adversary runs denoising techniques on a server with four NVIDIA RTX 3090 GPUs.

7.2 Security Analysis

We assess SecHeadset's security performance from two perspectives: 1) if the attacker can obtain the transmitted essential data in Sec. 6 and 2) if not, how much information can the attacker extract from the voice data.

7.2.1 Analysis of the data exchange protocol. In our protocol a random *RN* and a speaker id *s* is transmitted using public-key cryptography, which is considered secure in most cases. So we assume an attacker can only obtain *RN* and *s* through random guessing. Since *D* is finite, *s* is limited (e.g. 2484 in LibriSpeech), making *RN* the primary source of security. We assume a k-bits *RN* and the attacker can guess 10^9 times per second considering of supercomputer capabilities, the attacker needs $\frac{2^k}{10^9 \cdot 365 \cdot 24 \cdot 3600}$ years to get *RN*. With k=128, this takes decades, which could be considered secure and aligns the NIST SP 800-57 standard [18]. Besides, each guess requires to generate noise and perform voice retrieval for validation, further making the attack infeasible.

7.2.2 How much information can adversaries obtained. We evaluate three types of adversaries with different capabilities:

- *Type-A* does not know the detail of SecHeadset and extracts information from the obtained audio directly.
- *Type-B* does not know the detail of SecHeadset but will employ state-of-the-art (SOTA) speech enhancement methods [20] with pre-trained weights. We do not consider speech separation here due to lack of suitable pre-trained models for 4-channel audios.
- *Type-C* is the most powerful that knows details of SecHeadset and trains specialized denoising models to recover voices, similar to that in Sec. 5.1. We consider both speech enhancement and separation based on [20]. Please note that even for this powerful attacker, it is not feasible to generate the exact obfuscation noise series, as they cannot obtain the speaker ID *s* and the random seed *RN* for each communication session.

We consider two types of audio data that adversaries could obtain: Original audio that could be obtained by exploiting vulnerabilities in software. Distorted audio affected by channel distortions, acquirable by eavesdropping on the channel. Both VoIP and voice message channels are considered.

Adversary with original audios: In general, adversaries with original audios are more likely to obtain information, as our noise is purely additive thus more likely to be removed. However, Tab. 2 shows that even in this scenario, the WER of all adversaries exceeds 75%, indicating the robustness of our phoneme-based noise. The STOI of different adversaries is consistently low, suggesting a poor perception quality. The result also reveals that the pre-trained models (Type B) do not clarify the audio but make it harder to recognize, possibly due to the amplification of phonemes in our noise by these models. Among the four types of adversary, Type C with speech enhancement model is the most powerful.

Adversary	Type A	Type B	Type C-Enh	Type C-Sep	Clean voices
WER(%) ↑	95.27	98.83	76.75	81.90	3.070
STOI ↓	0.4617	0.2602	0.5895	0.5397	1.000

Table 2: Adversaries with original obfuscated audios.

Adversary with distorted audios: Results in the left two columns of Fig. 17 show that when without protection, attackers can recognize the audio with low WER (<10%) and high intelligibility (STOI>0.7). With the protection of SecHeadset, in the VoIP channel, the WER of all types of adversaries exceeds 85%. In the voice message channel where the channel distortion is smaller, the WER remains higher than 75%, suggesting that our noise still performs well. Additionally, the average STOI score of the audios obtained by adversaries is around 0.3 in the VoIP channel and 0.4 in the voice message channel, indicating poor intelligibility.

7.3 Usability Analysis

We transmit audios with SecHeadset through different communication channels, collect the output audio, recognize audios with ASR, and calculate the WER and the STOI score. For good usability, we aim for a low WER and a high STOI score. The third column of Fig. 17 shows the results of VoIP channels. Although the WER increases compared to that of the clean voices, for all applications except for WhatsAPP and WeChat, the WER<30%, with Skype even achieves 10%. The STOI of these six applications is approximately 0.6, indicating a reasonable intelligibility. We attribute the poor performance in WhatsAPP and WeChat to their overly stringent denoising algorithms during VoIP calls, which significantly distort the original voice component, thereby compromising the effectiveness of our voice retrieval process.

The fourth column of Fig. 17 illustrates the results of voice message channels. As the distortion is smaller than VoIP channel, our voice retrieval performs much better, achieving a WER comparable to the clear data in four applications (~5%). For WhatsAPP and WeChat, where our system performs poorly in the VoIP channel, it performs well in voice message channel, with WERs of 18.4% and 37.2%, respectively. The STOI score for all applications exceeds 0.5, with three applications even close to 0.75, demonstrating particularly robust performance. Besides, we find the WER and STOI score of Line is worse than that in the VoIP channel, which is contrary to other applications. Upon further investigation of cached audios in Line, we find that Line compresses voice messages into .aac format with low bitrates (17-25 kbps), which might be even lower than that in the VoIP channel, thereby limiting the usability of our system.

7.4 Comparison with Existing works

To our best knowledge, SecHeadset is the first voice obfuscation-based method for user-controlled secure voice communications. Existing works mainly focus on encryption-based methods. While many techniques exist, most are closed-source, making direct comparisons challenging. One exception is [16], which transmits encrypted audio features and uses a fine-tuned LPCNet [41] for audio reconstruction. Since all codes but the fine-tuned LPCNet weights are available, a direct performance comparison is not feasible. Instead, we compare the feature-level error percentage, focusing on the timbre feature, which is crucial for voice reconstruction. Assume the initial feature is $[f_{1,init}, f_{2,init}, ..., f_{n,init}]$, the decrypted feature is $[f_{1,dec}, f_{2,dec}, ..., f_{n,dec}]$, then the error percentage is calculated as $\frac{1}{n}\sum_{i=1}^{n}\frac{||f_{i,init}-f_{i,dec}||}{||f_{i,init}||}$ We compare SecHeadset and [16] across various distortions, including Opus lossy compression and 8 practical voice message channels. Results in Fig. 19 reveals that [16]

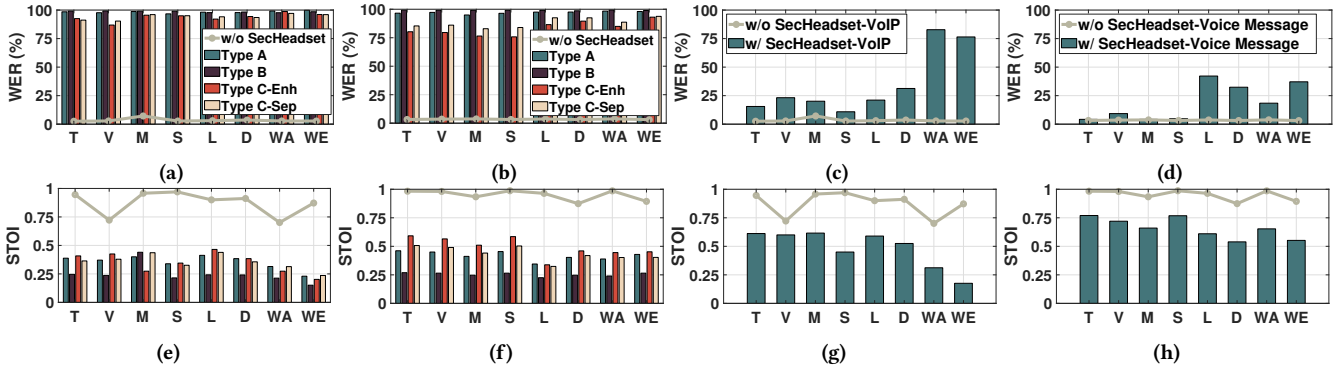

Figure 17: Results of the security and usability analysis. The line chart represents the scenraio without SecHeadset, where adversaries can obtain original voices in communication. Labels in x-axis represent applications, the same as Tab. 1. Figures in the same column share the same legend. (a)(e): Adversary in VoIP channel. (b)(f): Adversary in Voice Message channel. (c)(g): Usability analysis in VoIP channel. (d)(h): Usability analysis in voice message channel.

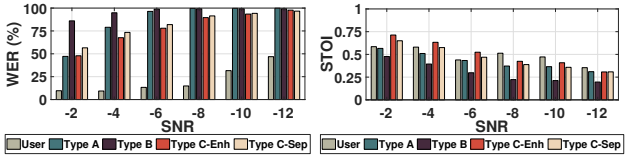

Figure 18: Impact of the noise energy level.

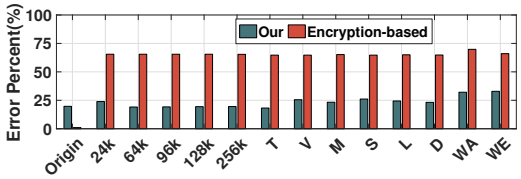

Figure 19: Feature error in SecHeadset and [16]. X-axis represents simulated distortions with opus lossy codec (24k-256k) or real distortions in VoIP channels (T-WE).

achieves near-zero error under ideal conditions, but degrades significantly (>50% error) under distortion. In contrast, SecHeadset maintains stable performance, with error rates remaining below 20% across all distortion conditions.

7.5 User Study

Ethics Considerations: Procedures involving human are validated by our Ethics Committee. The study is conducted in accordance with the Declaration of Helsinki [42] and the ICH guideline for good clinical practice [43]. All participants have signed an informed consent form before the study.

We recruit 24 participants (9 females and 15 males; aged 21-27) to rate audios. Inspired by MOS score [44], we ask participants to rate the clearness and intelligibility of the audio on a 1-5 scale, where 5 means the best quality and 1 means the worst. Here the clearness refers to the presence of noises and the intelligibility assesses the ease of understanding audio content. The reason for using a perception score instead of the recognition accuracy is to mitigate potential biases from audio content and participant's prior knowledge. We test eight types of audio: clean audio, clean audio transmitted via VoIP and voice message channels, audios obtained by different types of adversaries from VoIP channels, and audios retrieved from SecHeadset. We launch VoIP calls and voice message transmissions in smartphones with SecHeadset and collect these audios digitally. Participants are seated in a quiet room and using

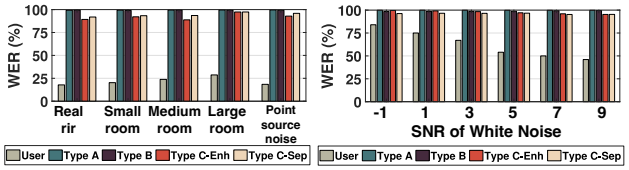

Figure 20: Impact of RIR and environmental noise.

a headband headset. Each of them needs to evaluate a total of $5 * 8 * 2 = 80$ audios.

Results in Tab. 3 show that the clearness and intelligibility of audios obtained by adversaries are poor, suggesting ineffective information extraction. In contrast, audio playback to users by SecHeadset exhibits much better quality. Although the clearness score of audios in VoIP channel is not very high (2.51), the intelligibility score is good (3.37), indicating users can effectively extract information from these audios. Notably, the voice message channel achieves intelligibility levels comparable to clean audio.

Type	Origin	VoIP	Voice Message	A-A
Clearness	4.95 ± 0.20	4.82 ± 0.28	4.93 ± 0.11	1.31 ± 0.40
Intelligibility	4.87 ± 0.34	4.84 ± 0.42	4.89 ± 0.34	1.43 ± 0.82

Type	A-C (Enh)	A-C (Sep)	Ours-VoIP	Ours-Voice Message
Clearness	1.72 ± 0.50	1.42 ± 0.28	2.51 ± 0.53	3.51 ± 0.40
Intelligibility	1.97 ± 0.77	1.69 ± 0.73	3.37 ± 0.80	4.27 ± 0.63

Table 3: Results of human perception. A-A and A-C refer to Attacker Type A and Attacker Type C.

7.6 Impact of Different Factors

7.6.1 Obfuscation Noise Energy. In the previous experiments, the energy of noise is fixed at a SNR of -9 dB. Here we vary the audio SNR from -12 dB to -2 dB, then measure the WER and STOI score for users and different adversaries. Results in Fig. 18 show that as the noise energy increases, the user's recognition accuracy first decreases slowly and then declines more rapidly when SNR<-8. While for adversaries, the WER first increases slowly and then stabilizes. For intelligibility, results show that the STOI scores of both adversaries and the user decrease gradually as the noise energy increases. Notably, at low noise levels (SNR>-8), the STOI score of adversary Type C is even higher than that of the users, which can be attributed to the good performance of the deep learning-based noise suppression techniques in high SNR scenarios. These

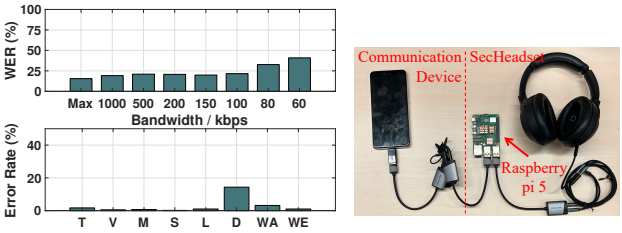

Figure 21: Left: impact of network bandwidth and data decoding error rate. Right: Portable Implementation.

results illustrate the privacy-usability tradeoff when choosing noise energy. Specifically, increasing noise energy could make it harder for adversaries to extract voice content thus enhance privacy, while it comes at the cost of communication quality for legitimate users, leading to worse usability. In practice, users can adjust the noise energy according to their privacy requirements.

7.6.2 RIR and Environmental Noise. To test the impact of room impulse response (RIR) and environmental noise, we add different types of RIRs and Gaussian white noise with varying energy to the clean audio. The audio is then transmitted through communication channels with `SecHeadset`. Results in the left of Fig. 20 show that the RIR causes an increase in the WER for users, but with a limited extent. Besides, the RIR also slightly increases the WER for adversaries, indicating better security. Additionally, as the room size increases, the WER gradually rises, which we attribute to stronger echoes. For the environmental noise, it impacts the usability significantly. We think this is caused by: 1) the environmental noise inherently affects the recognition accuracy, regardless of the usage of SecHeadset. 2) it affects the voice retrieval process. As the receiver is unaware of the environmental noise, the optimization process according to Equ. 2 tends to excessively reduce the noise and thus results in more noise residuals.

7.6.3 Network Bandwidth. We launch VoIP calls in Telegram under different network bandwidths. Results in Fig. 21 show that when the bandwidth > 90 kbps, the WER increases only slightly, indicating a limited impact on the usability of `SecHeadset`. When bandwidth falls below 90 kbps, the WER increases significantly, which corresponds with the results illustrated in Fig. 12.

7.6.4 Length of Probe Signal. We evaluate the impact of probe length on the accuracy of estimated $LR(f)$. The probe length varies from 0.5 to 3 seconds in 0.5-second increments. Results indicate that the error rate decreases from 81.2% at 0.5 seconds to 10.1% at 2 seconds, stabilizing around 7% thereafter. Thus, our 3-second probe is deemed appropriate for our scenario.

7.7 Portable Hardware Prototype

We implement a portable prototype of `SecHeadset` which costs about 75 dollars and consists of a development board (Raspberry Pi 5), a headset, and several soundcards, as shown in Fig. 21, The soundcards transmit audios between devices, the board runs the software prototype, and the headset records and plays audios. The software prototype is implemented using Python 3.11. The voice communication channel we tested is the Telegram VoIP channel.

Module	Algin.	STFT	Spec. Sub.	iSTFT	Res. Filter
Time (ms)	0.28	1.00	12.30	0.65	3.60

Table 4: Time consumption of voice retrieval process.

7.7.1 Computational Overhead. For real-time processing in practical usage, the time delay introduced by `SecHeadset` for each frame should be shorter than the frame length (64 ms in our design). Experiments show that the time consumed for processing each frame takes about 17.6 ms, much smaller than the frame length and suggests a real-time processing. The total extra delay introduced by `SecHeadset` is about 64 + 17.6 = 81.6 ms. Detail time consumption for each module of voice retrieval is shown in Tab. **??**.

7.7.2 Usability. We transmit audio samples in the test set over Telegram VoIP channel and collect audios playback by `SecHeadset` and then recognize them using ASR. The WER of the recognition results and the audio STOI score are 21.17% and 0.42, respectively, which performs worse than the software prototype running on a desktop (15.52% and 0.61). We attribute these performance decreases to the jitter and potential data loss when reading audio via the sound card in Raspberry Pi, which could cause misalignment and thus affect the voice retrieval process. This misalignment could also impact the STOI as the calculation of STOI requires precise alignment.

7.7.3 Data Error Rate. Fig. 21 shows the error rate of data transmission for each VoIP channel. Except for DingTalk and WhatsAPP which have error rates of 14.3% and 3.2%, the error rates of the others are all below 2%. The high error rate of the DingTalk VoIP channel could be attributed to its non-flat frequency response, as shown in Fig. 9.

8 Limitations & Future Directions

Poor usability in some VoIP channels. Currently, `SecHeadset` performs not well in WhatsAPP ans WeChat VoIP channels. We find the reason is that these channels treat obfuscated voices as noises and apply denoising procedures wrongly, making the voice irretrievable. To improve the usability, further channel-specific optimizations on obfuscation noise design to bypass noise detection in these channels is an important direction.

Unsatisfactory objective intelligibility. The objective intelligibility (STOI) of retrieved voices is relatively low in some VoIP channels. From results in Tab. 3 we notice that the subjective intelligibility is much higher than clearness, indicating that these audios are intelligible but contain residual noises. To further improve the objective intelligibility, applying lightweight deep learning-based speech enhancement techniques, such as DTLN [45], is a promising direction. DTLN has only two LSTM layers thus has the potential to be deployed on edge devices for real-time processing while maintaining good performance.

Poor performance under environmental noise. `SecHeadset` is sensitive to environmental noises, which affects our voice retrieval process (see Fig. 20). Pre-enhancing user voices before obfuscation with signal processing techniques or lightweight deep learning techniques like DTLN [45] is a potential solution.

Extra hardware requirement. `SecHeadset` currently runs on Raspberry Pi 5 and is implemented in Python, which is less efficient

than lower-level languages like C. Reimplementing with more efficient languages could make the system more lightweight, making it possible to be integrated into normal headsets or earbuds.

9 Discussion

Privacy-utility tradeoff: The design of SecHeadset inherently involves a privacy-utility tradeoff, which stems from several factors, including the energy level of obfuscation noise, the number of vowel series in noises, the voice retrieval method, and the channel state estimation technique. First, as shown in Fig. 8 and 18, increasing the noise energy or increasing the number of vowel series enhancing privacy but degrading voice quality. In the current implementation, a noise level of SNR = -9 dB and 3 vowel series are selected for better privacy. This parameters could be adjusted in usability-critical scenraios to improve voice quality. Second, SecHeadset employs signal processing-based methods for voice retrieval and channel state estimation, which reduce computational overhead at the cost of audio quality. In a scenario where audio quality in concerned and computational sources are sufficient, deep learning-based approaches could be used to improve the performance.

Built-in microphone-based eavesdropping: To our best knowledge, using built-in microphones of smart devices for eavesdropping is not feasible in our scenario. As stated in our system model (Sec. 3.1), we assume the hardware and operating system of the smart device are trustworthy. Under this assumption, adversaries are limited to use legitimate API calls for microphone access. However, the two most common mobile operating systems (Android and iOS) restrict such access [46–48]. For example, in iOS, audio recording requires either the AVAudioRecorder or AVAudioSession API. The former records from the system's active input device, which defaults to SecHeadset when connected. While the AvAudioSession API can configure microphone settings, it can only use one input device at a time. If it use the built-in microphone, the users voice would not be processed by SecHeadset, which would interrupt the communication and alert the user.

Assumption of Adversary's Knowledge: In the threat model in 3.2, we assume the adversary lack access to user-specific and communication session-specific information, i.e., the matched speaker s for each user and the random number RN. In different practical settings, it could be possible for adversaries to obtain these information. For example, obtaining s by social engineering and obtaining RN by exploiting vunerabilities in PKI. More investigations are needed in future works to understand the impact of these assumptions on SecHeadset's security performance.

Further In-depth Evaluations: Current evaluations are based on LibriSpeech. Since the effectiveness of obfuscation noises depends on the timbre similarity between the matched speaker s in dataset D and the user, if the user's voiceprint deviates the distribution of D, SecHeadset's performance may degrade. A more diverse and larger test set would provide better insights into SecHeadset's performance and robustness. Additionally, evaluations on different sizes of D would offer insights into the dataset requirements of SecHeadset.

Extendability to other languages: The phoneme-based noise generation scheme is based on the fundamental phonetic units (phonemes) of English. The existence of analogous phonetic units in other languages, either in the same language family (such as French and Spanish) or different families (such as Mandarin and Japanese), suggesting the potential for the extendability through the adaptation of noise generation algorithms to language-specific phonetic sets.

Other denoising methods: Currently we focus on deep learning-based denoising for adversaries. Other methods, such as correlation-based techniques, would be much challenging since: 1) The large database (~1000 hours) and misalignment make repetition almost impossible. 2) Even when receptive noises occur, variant user voices also introduce differences. 3) Even if adversaries can recover speech from receptive noise segments, these constitute only a small fraction of overall communications, limiting potential exposure.

User-perceived privacy: Despite SecHeadset's full transparency, non-expert users may struggle to understand its detailed operation, creating potential privacy concerns. User studies focusing on user-perceived privacy is needed to better understand users' concerns.

Handling channel distortion variations in a single communication. SecHeadset measures channel distortions only at session initiation. However, distortions could vary within one session, potentially affecting the voice retrieval process and reducing the voice quality. In our design, the frequency response (FR), lossy ratio (LR), and network conditions are the main factors. Fortunately, the FR and LR typically remain stable within each application due to codec specifications. For the variation of network conditions, a shorter segmentation size may help adapt to network variations.

10 Conclusion

In this paper, we propose SecHeadset, a practical, end-to-end, and plug-and-play system for voice privacy protection in real-time voice communication. The basic idea of our system is to obfuscate users' voices by adding carefully designed noise before transmission and retrieve the voices after receipt. We design a new type of phoneme-based noise for voice obfuscation and a lightweight spectral subtraction-based algorithm for voice retrieval. Furthermore, we design a protocol for real-time information exchange and channel distortion estimation, allowing our system to adapt to different communication channels without prior configuration. We evaluate our system on eight commonly used communication applications. Results show that our system effectively prevents adversaries from extracting information from transmitted audios, achieving a recognition accuracy below 15% without significantly sacrificing usability. We also implement SecHeadset with off-the-shelf portable hardware and verify its effectiveness in real-time voice communications.

Acknowledgments

We sincerely thank our anonymous reviewers and shepherd for their valuable feedbacks. This paper is supported in part by the National Key R&D Program of China (2023YFB2904000, 2023YFB2904001, and 2023YFB3107402), the Zhejiang Provincial Natural Science Foundation of China (LD24F020010 and LY24F020007), the National Natural Science Foundation of China (62172359 and 62472372), the Key Research and Development Program of Hangzhou City (2024SZD1A27), and the Key R&D Programme of Zhejiang Province (2025C02264).

References

[1] Market Growth Reports. Voice communication equipment market: Opportunities and forecast 2023-2029. https://www.marketgrowthreports.com/global-voice-communication-equipment-market-21331280, 2022.

[2] The Guardian News. Nsa monitored calls of 35 world leaders after us official handed over contacts. https://www.theguardian.com/world/2013/oct/24/nsa-surveillance-world-leaders-calls, 2013.

[3] David Rupprecht, Katharina Kohls, Thorsten Holz, and Christina Pöpper. Call me maybe: Eavesdropping encrypted LTE calls with revolte. In Srdjan Capkun and Franziska Roesner, editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 73–88. USENIX Association, 2020.

[4] BBC News. Pegasus: French president macron identified as spyware target. https://www.bbc.com/news/world-europe-57907258, 2021.

[5] CNN Business. Android apps are harvesting your data even after you tell them not to, says study. https://edition.cnn.com/2019/07/10/tech/android-apps-privacy-trnd/index.html, 2019.

[6] Kien Tuong Truong Jonas Hofmann. End-to-end encrypted cloud storage in the wild: A broken ecosystem. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS'24)*, Salt Lake City, USA, 2024. Association for Computing Machinery.

[7] Martin R. Albrecht, Sofia Celi, Benjamin Dowling, and Daniel Jones. Practically-exploitable cryptographic vulnerabilities in matrix. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, pages 164–181. IEEE, 2023.

[8] CVE-2020-25218. https://nvd.nist.gov/vuln/detail/CVE-2020-25218, 2020.

[9] CVE-2019-3568. https://nvd.nist.gov/vuln/detail/CVE-2019-3568, 2019.

[10] Wikipedia. Prism. https://en.wikipedia.org/wiki/PRISM, 2024.

[11] Hongjun Liu. Audio block encryption using 3d chaotic system with adaptive parameter perturbation. *Multim. Tools Appl.*, 82(18):27973–27987, 2023.

[12] Xinyu Ge, Guiling Sun, Bowen Zheng, and Ruili Nan. Fpga-based voice encryption equipment under the analog voice communication channel. *Inf.*, 12(11):456, 2021.

[13] Praveen Kumar Kasetty and Aniruddha Kanhe. Covert speech communication through audio steganography using DWT and SVD. In *11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020, Kharagpur, India, July 1-3, 2020*, pages 1–5. IEEE, 2020.

[14] Shambhu Shankar Bharti, Manish Gupta, and Suneeta Agarwal. A novel approach for audio steganography by processing of amplitudes and signs of secret audio separately. *Multim. Tools Appl.*, 78(16):23179–23201, 2019.

[15] Kasetty Praveen Kumar and Aniruddha Kanhe. Secured speech watermarking with dct compression and chaotic embedding using dwt and svd. *Arabian Journal for Science and Engineering*, 47(8):10003–10024, January 2022.

[16] Piotr Krasnowski, Jérôme Lebrun, and Bruno Martin. A novel distortion-tolerant speech encryption scheme for secure voice communication. *Speech Commun.*, 143:57–72, 2022.

[17] IETF Codec Working Group. Opus codec. https://opus-codec.org/docs/opus_api-1.5.pdf, 2024.

[18] Elaine Barker. *Recommendation for key management:: part 1 - general.* May 2020.

[19] Peng Huang, Yao Wei, Peng Cheng, Zhongjie Ba, Li Lu, Feng Lin, Yang Wang, and Kui Ren. Phoneme-based proactive anti-eavesdropping with controlled recording privilege. *IEEE Transactions on Dependable and Secure Computing (IEEE TDSC)*, 2024.

[20] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. Attention is all you need in speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 21–25. IEEE, 2021.

[21] Irwin Pollack. Auditory informational masking. *The Journal of the Acoustical Society of America*, 57(S1):S5–S5, April 1975.

[22] Telegram. https://telegram.org, 2024.

[23] Viber. https://www.viber.com/en/, 2024.

[24] Messenger. https://www.messenger.com, 2024.

[25] Skype. https://www.skype.com, 2024.

[26] Line. https://line.me, 2024.

[27] Dingtalk. https://www.dingtalk.com/en, 2024.

[28] Whatsapp. https://whatsapp.com, 2024.

[29] Wechat. https://www.wechat.com, 2024.

[30] Garofolo, John S., Lamel, Lori F., Fisher, William M., Pallett, David S., Dahlgren, Nancy L., Zue, Victor, and Fiscus, Jonathan G. Timit acoustic-phonetic continuous speech corpus, 1993.

[31] Stanley A. Gelfand. *Hearing: An Introduction to Psychological and Physiological Acoustics.* CRC Press, November 2017.

[32] Apple Inc. Additional tools for xcode. https://download.developer.apple.com/Developer_Tools/Additional_Tools_for_Xcode_15/Additional_Tools_for_Xcode_15.dmg, 2024.

[33] Antony W. Rix, John G. Beerends, Michael P. Hollier, and Andries P. Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2001, 7-11 May, 2001, Salt Palace Convention Center, Salt Lake City, Utah, USA, Proceedings*, pages 749–752. IEEE, 2001.

[34] Michael Chinen, Felicia S. C. Lim, Jan Skoglund, Nikita Gureev, Feargus O'Gorman, and Andrew Hines. Visqol v3: An open source production ready objective speech and audio metric. In *Twelfth International Conference on Quality of Multimedia Experience, QoMEX 2020, Athlone, Ireland, May 26-28, 2020*, pages 1–6. IEEE, 2020.

[35] International Telecommunication Union. *Synchronous frame structures used at primary and secondary hierarchical levels.* International Telecommunication Union, 1998. Retrieved from https://www.itu.int/rec/T-REC-G.704/en.

[36] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pages 5206–5210. IEEE, 2015.

[37] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L. Seltzer, and Sanjeev Khudanpur. A study on data augmentation of reverberant speech for robust speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pages 5220–5224. IEEE, 2017.

[38] Amazon transcribe. https://aws.amazon.com/transcribe/, 2024.

[39] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. A short-time objective intelligibility measure for time-frequency weighted noisy speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, 14-19 March 2010, Sheraton Dallas Hotel, Dallas, Texas, USA*, pages 4214–4217. IEEE, 2010.

[40] Vention usb external stereo sound card. https://www.amazon.com/dp/B08C55N34G/, 2024.

[41] Jean-Marc Valin and Jan Skoglund. LPCNET: improving neural speech synthesis through linear prediction. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 5891–5895. IEEE, 2019.

[42] *JAMA*, 310(20):2191, November 2013.

[43] International Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use (ICH). *ICH Guideline for Good Clinical Practice E6(R2)*, November 2016. Retrieved from https://www.ich.org/page/efficacy-guidelines.

[44] International Telecommunication Union. *Methods for Subjective Determination of Transmission Quality (ITU-T Recommendation P.800).* International Telecommunication Union, 1996. Retrieved from https://www.itu.int/rec/T-REC-P.800.

[45] Nils L. Westhausen and Bernd T. Meyer. Dual-signal transformation LSTM network for real-time noise suppression. In Helen Meng, Bo Xu, and Thomas Fang Zheng, editors, *21st Annual Conference of the International Speech Communication Association, Interspeech 2020, Virtual Event, Shanghai, China, October 25-29, 2020*, pages 2477–2481. ISCA, 2020.

[46] Apple Developer. AVAudioRecorder. https://developer.apple.com/documentation/avfaudio/avaudiorecorder, 2025.

[47] Apple Developer. Apple Session Programming Guide. https://developer.apple.com/library/archive/documentation/Audio/Conceptual/AudioSessionProgrammingGuide/OptimizingForDeviceHardware/OptimizingForDeviceHardware.html#//apple_ref/doc/uid/TP40007875-CH6-SW4, 2017.

[48] Android Developers. AudioManager. https://developer.android.com/reference/android/media/AudioManager, 2025.