Towards Reactive Acoustic Jamming for Personal Voice Assistants

Peng Cheng
Lancaster University
p.cheng2@lancaster.ac.uk

Ibrahim Ethem Bagci
Lancaster University
i.bagci@lancaster.ac.uk

Jeff Yan
Linköping University
jeff.yan@liu.se

Utz Roedig
Lancaster University
u.roedig@lancaster.ac.uk

ABSTRACT

Personal Voice Assistants (PVAs) such as the Amazon Echo are commonplace and it is now likely to always be in range of at least one PVA. Although the devices are very helpful they are also continuously monitoring conversations. When a PVA detects a wake word, the immediately following conversation is recorded and transported to a cloud system for further analysis. In this paper we investigate an active protection mechanism against PVAs: reactive jamming. A Protection Jamming Device (PJD) is employed to observe conversations. Upon detection of a PVA wake word the PJD emits an acoustic jamming signal. The PJD must detect the wake word faster than the PVA such that the jamming signal still prevents wake word detection by the PVA. The paper presents an evaluation of the effectiveness of different jamming signals. We quantify the impact of jamming signal and wake word overlap on jamming success. Furthermore, we quantify the jamming false positive rate in dependence of the overlap. Our evaluation shows that a 100% jamming success can be achieved with an overlap of at least 60% with a negligible false positive rate. Thus, reactive jamming of PVAs is feasible without creating a system perceived as a noise nuisance.

CCS CONCEPTS

• **Security and privacy** → *Access control*; *Systems security*; *Privacy protections*;

KEYWORDS

Reactive Acoustic Jamming; Wake Word Detection; Acoustic Privacy; Security and Privacy in IoT

ACM Reference Format:
Peng Cheng, Ibrahim Ethem Bagci, Jeff Yan, and Utz Roedig. 2018. Towards Reactive Acoustic Jamming for Personal Voice Assistants. In *Proceedings of ACM Conference (MPS'18).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Personal Voice Assistants (PVAs) are deployed as stand alone devices such as Amazon Echo or Google Home, are integrated within every phone, tablet and PC (Siri, Cortana), are used in appliances such as TVs and set-top boxes (LG, SKYQ) and are integrated into cars (Mercedes). Thus, it is very likely that we are always at least in range of one PVA.

Using microphones, PVAs are monitoring the acoustic channel for a specific spoken word. Once this *wake word* has been detected the PVA records the immediately following audio signal which is then transported to a back-end system for further analysis. The back-end system then analyses the audio signal and extracts spoken user commands.

Out of security and privacy concerns, people would like to be in control of surrounding PVAs, for example, to disable the ability to issue commands or to prevent recording of conversations. When a person is in control of the PVA she can simply turn off the device. However, in environment such as public spaces this is impossible.

In this paper we investigate *acoustic reactive jamming* as a method of disabling PVAs. A Protection Jamming Device (PJD) is used which emits an acoustic jamming signal to prevent a PVA from analyzing audio signals. However, instead of continuously jamming the channel, which would be perceived as continuous noise nuisance, jamming is applied reactively at specific moments. Specifically, the PJD recognizes the spoken wake word and applies a jamming signal to it. Thus, the PVA fails to recognize the wake word and it does not record the following audio signal.

The PJD will also be useful for other protection scenarios which are beyond the scope of this paper. For example, recent work has shown that PVA can be triggered by attackers using inaudible voice commands (see Zhang et al. [25]) which could be recognized and jammed. Our focus is on designing a protection device, empowering people to control PVAs tapping into their conversations. It also has to be noted that the described mechanism can be exploited by a nefarious actor to disable a PVA.

Signal jamming has been previously employed as effective protection mechanism. For example, reactive jamming has been used to protect wireless communication networks [2, 14]. The packet header containing source and destination addresses is evaluated and, if required, a jamming signal is applied to the remainder of the packet, preventing packet reception. Our work transfers this reactive jamming method to the acoustic domain. A recent work by Roy et al. [17] demonstrates inaudible jamming in the acoustic domain. Non-linearity of the microphone shifts the white noise jamming signal in the ultrasound frequency range to the audible range. This work shows how to jam an acoustic system without people noticing the signal directly. However, this existing work is not tailored to the PVA context and uses continuous jamming signals which are inefficient and also might constitute a potential health risk. To the best of our knowledge, the work presented in this paper is the first investigating reactive jamming targeting PVAs.

Reactive jamming requires two elements: *wake word detection* and *jamming*. The protection device must be able to detect the wake

MPS'18, October 2018, Toronto, Canada

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

word faster than the PVA. The protection device can then apply the jamming signal to the later section of the wake word. If the overlap is sufficient, wake word detection by the PVA is prevented. The jamming signal must be effective and people should not consider it as noise nuisance. Thus, the signal should only be applied when needed (low false positive rate), be not too loud and somewhat pleasant to listen to.

This paper provides two specific contributions. First, we quantify the impact of jamming signal and wake word overlap on the jamming success. We use four different wake words used by the popular PVA Amazon Echo and four different jamming signals for our study. Our evaluation shows that PVA wake words can be jammed with a 100% success rate in most cases when jamming signal (Additive White Gaussian Noise (AWGN)) and wake word overlap more than 60%. Second, We quantify the jamming false positive rate in dependence of the required overlap. The protection device has to recognize the keyword faster than the PVA leading to false positives; jamming is applied when not required. Our evaluation here shows that false positive rates are negligible.

2 PERSONAL VOICE ASSISTANTS

Many companies nowadays have their own signature Personal Voice Assistant (PVA) software and hardware. For instance, Apple appliances integrate Siri, Amazon provides Alexa, Google integrates the Google voice assistant, and Samsung gadgets work with Bixby.

2.1 System Overview

There are two operation phases of a PVA: *activation phase* and *recognition phase*. In the activation phase, a user needs to activate the PVA to initiate speech recognition. Typically, a user presses a specific button (e.g. as used on a SKYQ remote) or simply says a wake word (e.g. *Alexa* in case of Amazon's Echo). Most systems implement a wake word as this improves usability. The wake words may be speaker-dependent (e.g. *Hey Siri*) or speaker-independent (e.g. *Alexa*) [25]. Wake word recognition is continuously active on PVA devices. Once the key word is detected, the PVA enters the recognition phase. For most systems the audio signal following the wake word is streamed to a back-end cloud service for analysis. The cloud service is used to analyze the captured audio signal to extract user commands. It also may store captured audio signals.

Central part of a PVA device is the wake word recognition implementation. Cooperations such as Apple or Microsoft do not provide details of their implementations; however, open source toolkits such as AlexaPi based on PocketSphinx developed by CMU [5] are available. All major wake word recognition implementations have similar performance from users' perception and are based on only several major approach options.

Speech Recognition (SR) is employed for wake word detection in the activation phase on the device and as well for command recognition on the back-end during the recognition phase. SR for wake word recognition can be less complex compared to the one used in the back-end as only one or a few words must be recognized. Wake word recognition is a specific application of SR referred to as Keyword Spotting (KWS) in the literature. Often, wake word recognition is implemented in dedicated hardware to improve energy consumption of battery powered devices. Wake words can be speaker-dependent and in this case the legitimate user has to train the PVA.

2.2 Speech Recognition and Keyword Spotting

The process of SR begins with separation of the audio input based on pauses. Each identified speech block is called an utterance which may be a word or a non-linguistic sound (e.g. cough, um, breath) [4]. Each utterance is then split into segments. A feature vector is extracted to represent each unit.

Models need to be constructed to predict what language elements these units represent. Gaussian Mixture Models (GMMs) are the commonly used acoustic models. However, recently these have been replaced by models trained by Deep Neural Network (DNN) as they are more robust. These models can tolerate better environmental and hardware specific variations [9, 13, 15].

Further processing is necessary to deal with temporal variability [9]. Hidden Markov Models (HMM) are normally applied to Automatic Speech Recognition (ASR) and KWS. Some recent KWS solutions can also work without HMM [3, 13].

Other techniques using Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs)/Long Short-Term Memorys (LSTMs) instead of the combination of DNN and HMM for KWS also exist [13].

Apart from these techniques, some KWS systems use Large Vocabulary Continuous Speech Recognition Systems (LVCSR) to decode audio and generate lattices [3, 13], then they can be used for indexing and keyword searching. These systems focus on large audio database applications rather than audio streaming applications, which is outside the scope of this paper.

2.3 PocketSphinx

AlexaPi based on PocketSphinx performs wake word recognition using the GMM-HMM approach for KWS described earlier [10]. PocketSphinx [10] is the optimized version of CMU's SPHINX (an open source LVCSR system) for resource limited embedded systems. PocketSphinx provides wake word selection and detection threshold tuning functions.

Products such as Amazon Echo/Echo Dot or Google Home use proprietary algorithms (often in combination with specialist hardware) based on more recent techniques discussed in the previous section to perform KWS. For example, Amazon products mainly use DNN-HMM solutions, Google and other vendors use solutions such as a single DNN followed by a posterior handling method.

Although PocketSphinx doesn't apply state-of-the-art modeling technique, it is still a reliable KWS solution [7, 11]. Because it is an open source speech recognition system it is often used instead of proprietary speech recognition toolkits. In particular, small companies or independent developers make use of PocketSphinx.

In our evaluation we use AlexaPi based on PocketSphinx, which means at this stage we only focus on jamming wake word recognition based on GMM-HMM. We treat the wake word recognition as a black box, meaning that our jamming signals are not designed to the specifics of the wake word recognition algorithm.

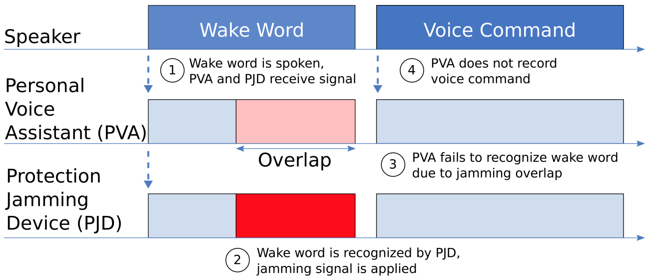


Figure 1: Wake word jamming framework. (1) The wake word is received by PVA and PJD. (2) The PJD detects the wake word faster than the PVA and applies jamming. (3) The PVA fails to recognise the wake word. (4) The PVA does not record the voice command.

2.4 Security Considerations

It is probable that the voice data sent to the back-end and also extracted commands are stored. Companies such as Amazon and Google can use the data in many ways. The command history extracted from audio streams can be used for marketing purposes (learning user behavior). The audio streams can be used as training data for the SR system. Companies are not very clear in regards to how captured data is processed and used. A user must therefore assume that recorded audio streams or results from processing these remain in the back-end system.

Users might trigger the PVA unintentionally by using the wake word (e.g. by talking about your friend Alexa). In this case a conversation that is not a command to the PVA is recorded. Users might want to be in control, and ensure that wake word recognition is disabled. If users are not device owners this is difficult to achieve (e.g. in public spaces with PVA systems). Wake word jamming as described in this paper can help in such situation.

Another threat is from adversaries who are familiar with the working mechanism of PVA and are skilled at producing human unnoticeable or human inaudible sound attacks. Some research [18, 25] has shown how to send human inaudible but machine understandable voice commands to attack PVAs. Our work in this paper sheds light on how to protect against such covert attacks.

An attacker might also use the jamming technique we explore in this paper to prevent the use of PVA systems. A jamming device using a human inaudible reactive jamming signal will be very hard to detect and will prevent the use of any PVA in the vicinity. An attacker might further use jamming to hijack the system. The attacker can prevent the PVA to execute user commands and instead perform an action. The user has the impression of interacting with the system while another device is acting on its behalf.

3 PERSONAL VOICE ASSISTANT JAMMING

It is our aim to design a reactive jamming framework which can be used to intentionally disable PVAs. We aim to interrupt the wake word detection in the activation phase so that the PVA never reaches the recognition phase. Jamming is applied selectively, preventing continuous jamming signals that can be perceived as noise nuisance or even have health implications. The jamming framework can be instantiated to become a Protection Jamming Device (PJD).

3.1 Jamming Approach

The overall design of the jamming approach is shown in Figure 1:

(1) PVA and PJD are continuously observing the acoustic channel. Both devices listen to the channel and try to determine if the wake word is spoken.
(2) The PJD is set to detect the wake word earlier than the PVA by only looking for the first part of it. Upon detection, the PJD applies the jamming signal.
(3) The jamming signal now overlaps with the remainder of the wake word currently still analyzed by the PVA. The jamming signal prevents recognition of the wake word by the PVA.
(4) The following voice command is not recorded by the PVA as it does not transition from activation to recognition phase.

The outlined approach is possible as the PJD is set to detect the wake word much faster than the PVA. Thus, enough time is available for the PJD to apply an effective jamming signal. The time duration in which jamming signal coincides with the later part of the wake word is referred to as *overlap*.

3.2 Design Challenges

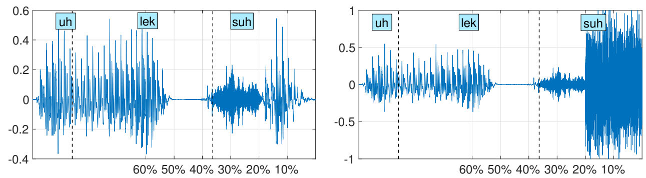
A PVA's wake word recognition is designed to be very accurate to prevent accidental triggering. We exploit this fact for the design of the PJD and deliberately sacrifice accuracy for detection speed. The PJD is set to react to the start of the wake word and triggers jamming when there is a match. Obviously, the PJD will be less accurate in wake word recognition than the PVA and false jamming will be the result. It is the aim to jam as effective as possible which means that the overlap should be maximized. However, the overlap should be minimized to prevent unnecessary jamming. An overlap should be found that balances these optimization goals.

Reliable jamming requires a jamming signal that interferes with recognition of the target signal as much as possible. Intuitively, the jamming signal should be as strong as possible and it should also overlap with the target signal as much as possible. However, the design of the interfering signal is also important. The signal should be not perceived as noise nuisance and, thus, the most effective jamming signals might not be usable. Furthermore, the signal shape could be tailored to the SR algorithms employed. This approach might result in a very effective jamming signal but will only be effective for the specific SR system. Finally, the jamming signal could take the signal shape of the keyword into account to effectively jam its specific properties.

3.3 Jamming Evaluation Framework

In our evaluation, described in the next sections, we explore and evaluate the aforementioned design space. We use the following evaluation framework.

We used an open source software called AlexaPi [1] running on a Raspberry Pi 3 Model B. The AlexaPi uses an open source SR system called Pocketsphinx developed by CMU [5] to detect wake words. The system uses the Amazon cloud service as back-end system for following speech analysis. The system is similar to commercial Amazon products such as the Amazon Echo; the difference is the implementation of the wake word recognition. The Raspberry Pi together with the AlexaPi and Amazon cloud service is the PVA system used in our experiment.



(a) Alexa **(b) Alexa with AWGN**

Figure 2: The audio signal of the wake word *Alexa* without and with 20% overlap of the AWGN jamming signal.

As we aim to evaluate systematically jamming performance we create the audio input signal for the PVA system artificially. This input signal contains the wake word and, with a set overlap, the jamming signal. The generated combination of wake word and jamming signal is directly fed into the AlexaPi via a virtual microphone input. By bypassing speakers, microphones and other system elements we can study accurately the impact of jamming signal, overlap and Signal-to-noise Ratio (SNR) without system related influences.

The wake word voice for the audio input signal is generated using the open source Text to Speech (TTS) software Festival [6]. This method provides us with a standardized signal that can be reproduced[1]. The wake word is then mixed with the chosen jamming signal using Matlab.

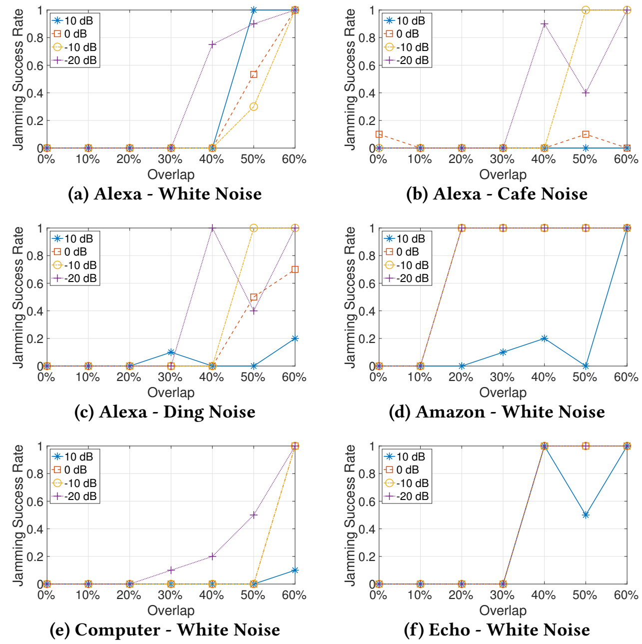
4 JAMMING EVALUATION

We aim to quantify the impact of jamming signal and wake word overlap on the jamming success. We evaluate the effectiveness of audible jamming signals first; thereafter we investigate the practicality of inaudible jamming. Wake word variants, noise signal types, SNR and overlap are the four parameters varied in the experiments.

We use the wake words *Alexa*, *Amazon*, *Echo* and *Computer* because these are wake word options for Amazon Echo products.

In the first set of experiments, we chose AWGN, a *Ding*, and a short audio recording of ambient noise in a *Cafe* as noise signals (Audible Jamming). AWGN is known to have good interference properties while it is not a pleasant noise for users; the *Ding* and *Cafe* noise is less intrusive but has potentially less jamming capabilities. SNR levels of 10dB, 0dB, -10dB and -20dB are used. An overlap from 0% to 60% is selected.

In the second set of experiments, we use inaudible jamming signals. An AWGN signal is created and then a high pass filter is used to filter out components below 20kHz. In practice, the signal is still audible as perfect filtering is not achievable (The signal is limited in the time domain). When the SNR is 10dB, 0dB and -10dB, the signal is barely noticeable. However, when the SNR reaches -20dB, the noise is obvious. Thus, a second noise signal with a band pass between 22kHz and 24kHz is used which is less audible as it has less frequency leakage in the lower frequency range. Again, an overlap from 0% to 60% is selected.



(a) Alexa - White Noise **(b) Alexa - Cafe Noise**

(c) Alexa - Ding Noise **(d) Amazon - White Noise**

(e) Computer - White Noise **(f) Echo - White Noise**

Figure 3: Average jamming success rates when using audible jamming signals. The AWGN (White Noise) is the most successful jamming signal. The required jamming overlap is wake word dependent. An overlap of more than 60% achieves a 100% jamming success for all wake words.

4.1 Audible Jamming

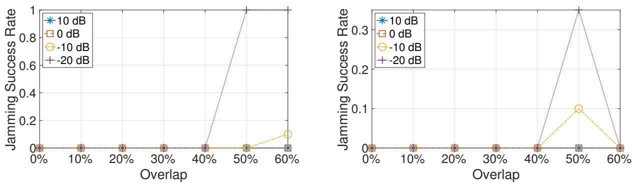
Alexa is used as wake word. The audio signal in time domain is shown in Figure 2a which also depicts the range for each syllable in the wake word. Figure 2b shows the same audio signal with added AWGN noise overlapping 20%.

Figure 3a shows the result of jamming *Alexa* with AWGN. The x-axis represents the overlap; how much of the signal was jammed, counting from the end of the wake word. The y-axis indicates the jamming success rate. For each SNR a different curve is included. Each data point is created by executing the experiment 10 times.

Figure 3a shows, as one would intuitively expect, that a larger overlap results in a better jamming success. Also, as expected, the strongest jamming signal with an SNR of -20dB is the most effective (with 40% overlap this signal can jam with a success rate of 75%). However, when reaching a 50% overlap there is no consistent correlation between SNR and jamming success. For example, the 10dB signal can jam successfully with an overlap of 50% while signals with an SNR of 0dB and -10dB are less successful (53% and 30%). We suspect that this behavior is due to the specific structure of the wake word; the *k* sound in *Alexa* is similar to the interference signal used for jamming which sits at 50%. When the jamming length reaches 60%, jamming with all four SNR levels is 100% successful.

Results of jamming Alexa with a Cafe noise is shown in Figure 3b. As the jamming length increases, jamming success rate is negligible for SNR of 10dB and 0dB. For SNR of -10dB and -20dB, the

[1]The exception was the word *Computer* for which we used a human voice recording; the software was unable to synthesize the correct pronunciation of the *C*.



(a) 20-24kHz **(b) 22-24kHz**

Figure 4: Average jamming success rates when using inaudible jamming signals. Inaudible jamming of wake words is feasible.

jamming rate is higher. Results of jamming *Alexa* with Ding noise (see Figure 3c) has similar results except higher jamming success rate for 0dB SNR.

From these experiments it is clear the AWGN is the best jamming signal. It is also shown that with an overlap of more than 60%, jamming is 100% successful, even when using a very quiet jamming signal of only 10dB.

AWGN is used to jam the different wake words *Amazon*, *Computer* and *Echo*. The results are depicted in Figure 3d, Figure 3e and Figure 3f. A noise with an SNR of 10dB can jam reliably the *Amazon* wake word with an overlap of more than 60%. However, once the SNR is less than 10dB, jamming is effective with an overlap of 20% or more. The wake word *Computer* requires an overlap of more than 60% for reliable jamming. *Echo* can be jammed with an overlap of 40%, with the exception of an SNR of 10dB where a jamming success of only 50% is achieved.

This evaluation shows that the required overlap is dependent on the wake word. However, it is also shown that an overlap of 60% is sufficient in most cases which gives a PJD enough time to apply the jamming signal.

4.2 Inaudible Jamming

Using the wake word *Alexa* the effectiveness of the two inaudible jamming signals is investigated. The results of this experiment are shown in Figure 4a and Figure 4b.

With the 20kHz-24kHz jamming signal and an SNR of -20dB and an overlap above 50% reliable jamming is achieved. With the 22kHz-24kHz signal reliable jamming is achieved with a 50% overlap. The result here is interesting as it is possible to jam the system with a noise signal that exists outside the spectrum that voice occupies.

To this end we can only speculate why this approach is possible; we offer three explanations: (i) The wake word recognition algorithm extracts features from the spectrum; the inaudible frequency range below 24kHz may be included. (ii) The frequency leakage in the audible frequency range is sufficient to affect the recognition process. (iii) The PocketSphinx integrates a software Automatic Gain Control (AGC) and the low frequency voice signal is reduced as the gain is adjusted to the high frequency noise.

The experiments show that there is potential to develop a jamming approach that makes use of jamming signals that are inaudible to humans and are therefore non intrusive.

5 FALSE POSITIVE JAMMING EVALUATION

The PJD may trigger unnecessary jamming as it must recognize the keyword before the PVA. With the time window available to the PJD, words with similar beginning to the wake word may trigger jamming. These *false positive* jamming events are not desirable as they introduce unnecessary noise nuisance.

Wake Word	Searched Letters	Frequency (%)	Overlap
Alexa	ale*	0.0014	52%
Amazon	am*	0.0896	56%
	em*	0.0625	
	Total	0.1585	
Computer	com*	0.2832	74%
Echo	ech*	0.00042	47%
	ek*	0.000061	
	ak*	0.00025	
	ach*	0.00292	
	Total	0.00798	

Table 1: Frequency of the words that start with pronunciation similar to the keywords. The spoken word data is taken from British National Corpus 2014 [12]

We investigate the false positive jamming attempts by looking at the frequencies of words in a spoken word corpus that start with a pronunciation similar to the wake words. We use the British National Corpus 2014 [12] consisting of 11,422,617 words. We search for the words that start with similar pronunciation to the wake words. For example, we consider the words starting with *ale* for the keyword *Alexa*. It should be noted that some words starting with *ale* may not be pronounced similar to Alexa (for example, the word *ale* itself). Thus, the results represent a worst-case analysis.

Table 1 shows the results. If we consider *ale**, 0.0014% of commonly spoken words are similar to *Alexa*. The overlap for jamming in this case is 52%. If the PJD uses *ale** as trigger, the last 52% of the wake word can be jammed and we would expect a false positive jamming for 0.0014% of spoken words. The overlap here is according to syllables boundaries and an analysis of overlap values of exactly 50% or 60% is not sensible.

We believe that false positive rates are acceptable for a practical jamming device, especially if the jamming signal is in the inaudible frequency space.

6 RELATED WORK

Jamming is a well studied subject in the communication domain. Existing jamming work focus either on disruption of communications [16, 21, 24] or implementation of novel protection mechanisms [19].

Reactive jamming has been used to protect wireless communication networks [2, 8, 14, 19, 20, 22, 23]. The packet header is evaluated and, if required, a jamming signal is applied to the remainder of the packet, preventing reception. Our work is similar but we apply this concept to acoustic signals.

There are few work aiming at jamming-based protection in the acoustic domain. Roy et al. use inaudible jamming against eavesdropping [17]. Although it is not reactive jamming, we can make use of the reported approach to inaudible jamming.

7 CONCLUSION

Our work shows that reactive jamming of PVAs wake words is a feasible approach. The approach can be used for protection, to control when PVAs can function. However, the mechanism could also be exploited by an attacker to block PVA services.

We have demonstrated that modestly strong audio signals with 10dB SNR and an overlap of 60% (with AWGN) can block wake word detection with a 100% success rate in most cases. This means that the PJD has at least 40% of the wake word duration to make a jamming decision. We have shown that this may lead to a false jamming; however, the false jamming rate is very small and should be acceptable for most practical scenarios. We have also shown that it is feasible to move the jamming signal into the inaudible frequency range, making it more applicable.

Our next steps are to carry out an evaluation with off-the-shelf PVAs and to supply audio and jamming signals via speakers instead of directly supplying generated audio signals to the PVA. We also plan to improve the design of inaudible jamming signals and to construct a practical PJD.

REFERENCES

[1] AlexaPi. 2016. Alexa client for all your devices! https://github.com/alexa-pi/AlexaPi. (2016).

[2] James Brown, Ibrahim Ethem Bagci, Alex King, and Utz Roedig. 2013. *Defend Your Home! Jamming Unsolicited Messages in the Smart Home.* ACM Press, 1–6. https://doi.org/10.1145/2463183.2463185

[3] Guoguo Chen, Carolina Parada, and Georg Heigold. 2014. Small-footprint keyword spotting using deep neural networks.. In *ICASSP*, Vol. 14. Citeseer, 4087–4091.

[4] CMUSphinx. 2006. Basic concepts of speech recognition. https://cmusphinx.github.io/wiki/tutorialconcepts/. (2006).

[5] CMUSphinx. 2006. Open Source Speech Recognition Toolkit. https://cmusphinx.github.io/. (2006).

[6] Festival. 2004. The Festival Speech Synthesis System (version 2.4). http://www.cstr.ed.ac.uk/projects/festival/. (2004).

[7] Christian Gaida, Patrick Lange, Rico Petrick, Patrick Proba, Ahmed Malatawy, and David Suendermann-Oeft. 2014. Comparing open-source speech recognition toolkits. *Tech. Rep., DHBW Stuttgart* (2014).

[8] Shyamnath Gollakota, Haitham Hassanieh, Benjamin Ransford, Dina Katabi, and Kevin Fu. 2011. They can hear your heartbeats: non-invasive security for implantable medical devices. In *ACM SIGCOMM Computer Communication Review*, Vol. 41. ACM, 2–13.

[9] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine* 29, 6 (Nov 2012), 82–97. https://doi.org/10.1109/MSP.2012.2205597

[10] David Huggins-Daines, Mohit Kumar, Arthur Chan, Alan W Black, Mosur Ravishankar, and Alexander I Rudnicky. 2006. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, Vol. 1. IEEE, I–I.

[11] Patrick Lange and David Suendermann-Oeft. 2014. Tuning Sphinx to outperform GoogleâĂŹs speech recognition API. In *Proc. of the ESSV 2014, Conference on Electronic Speech Signal Processing*. 1–10.

[12] Robbie Love, Claire Dembry, Andrew Hardie, Vaclav Brezina, and Tony McEnery. 2017. The Spoken BNC2014. *International Journal of Corpus Linguistics* 22, 3 (2017), 319–344.

[13] Minhua Wu; Sankaran Panchapagesan; Ming Sun; Jiacheng Gu; Ryan Thomas; Shiv Naga Prasad Vitaladevuni; Bjorn Hoffmeister; Arindam Mandal. 2018. Monophone-Based Background Modeling for Two-Stage On-Deivce Wake Word Detection. (2018). http://sigport.org/2800

[14] Ivan Martinovic, Paul Pichota, and Jens B. Schmitt. 2009. Jamming for Good: A Fresh Approach to Authentic Communication in WSNs. In *Proceedings of the Second ACM Conference on Wireless Network Security (WiSec '09)*. ACM, New York, NY, USA, 161–168. https://doi.org/10.1145/1514274.1514298

[15] Sankaran Panchapagesan, Ming Sun, Aparna Khare, Spyros Matsoukas, Arindam Mandal, Björn Hoffmeister, and Shiv Vitaladevuni. 2016. Multi-Task Learning and Weighted Cross-Entropy for DNN-Based Keyword Spotting.. In *INTERSPEECH*. 760–764.

[16] Konstantinos Pelechrinis, Marios Iliofotou, and Srikanth V Krishnamurthy. 2011. Denial of service attacks in wireless networks: The case of jammers. *IEEE Communications surveys & tutorials* 13, 2 (2011), 245–257.

[17] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. 2017. BackDoor: Making Microphones Hear Inaudible Sounds. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17)*. ACM, New York, NY, USA, 2–14. https://doi.org/10.1145/3081333.3081366

[18] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Inaudible Voice Commands: The Long-Range Attack and Defense. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Renton, WA, 547–560. https://www.usenix.org/conference/nsdi18/presentation/roy

[19] Matthias Schulz, Francesco Gringoli, Daniel Steinmetzer, Michael Koch, and Matthias Hollick. 2017. Massive reactive smartphone-based jamming using arbitrary waveforms and adaptive power control. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 111–121.

[20] Wenbo Shen, Peng Ning, Xiaofan He, and Huaiyu Dai. 2013. Ally friendly jamming: How to jam your enemy and maintain your own wireless connectivity at the same time. In *Security and Privacy (SP), 2013 IEEE Symposium on*. IEEE, 174–188.

[21] Matthias Wilhelm, Ivan Martinovic, Jens B Schmitt, and Vincent Lenders. 2011. Short paper: reactive jamming in wireless networks: how realistic is the threat?. In *Proceedings of the fourth ACM conference on Wireless network security*. ACM, 47–52.

[22] Wilhelm, Matthias and Martinovic, Ivan and Schmitt, Jens B and Lenders, Vincent. 2011. WiFire: a firewall for wireless networks. In *ACM SIGCOMM Computer Communication Review*, Vol. 41. ACM, 456–457.

[23] Fengyuan Xu, Zhengrui Qin, Chiu C Tan, Baosheng Wang, and Qun Li. 2011. IMDGuard: Securing implantable medical devices with the external wearable guardian. In *INFOCOM, 2011 Proceedings IEEE*. IEEE, 1862–1870.

[24] Wenyuan Xu, Wade Trappe, Yanyong Zhang, and Timothy Wood. 2005. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 46–57.

[25] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible Voice Commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*. ACM, New York, NY, USA, 103–117. https://doi.org/10.1145/3133956.3134052