Measuring How Much Amazon Echo Uploads via Traffic Analysis

Enze Liu, TJ Smith, Zesen Zhang
University of California, San Diego
{e7liu,tjs003,zez003}@eng.ucsd.edu

ABSTRACT

Smart speakers, such as the Amazon Echo, have gained popularity in recent years. These smart speakers normally have always-on microphones that are used to detect voice commands. While very useful, this behavior of constantly listening has raised a wide range of privacy concerns. These include, but are not limited to, what is being recorded, how the collected data is used and stored, and whether it is being protected well. In this work, we quantitatively answer two frequently asked privacy-related questions about Amazon Echo. First, does it stream any conversation before it is activated? Second, is any audio being sent after the end of a command? We designed and performed measurements that allowed us to answer these two questions. We first confirm that usually Echo is not transmitting any audio that happens before it is activated. We further verify that in the case of correctly detecting the end of a command, Echo will not stream any conversation that occurs after the command. Additionally, we are able to quantify that a 0.7 s gap is needed for Echo to correctly detect the end of a command. While there is a rich literature on analyzing Echo's network traffic, ours is the first to use its network traffic to answer the two aforementioned questions in a quantitative manner.

1 INTRODUCTION

Physical devices connected to the Internet, also known as Internet-of-things (IoT) devices, have gained popularity in recent years. Among numerous types of IoT devices, smart speakers with voice assistants, such as the Amazon Echo and the Google Home, are widely seen in users' homes. These devices detect and respond to voice commands. They normally have always-on microphones, which theoretically start recording after hearing a wake word and then send the voice data to a server via Internet for further processing [4].

This behavior has sparked many privacy concerns, including but not limited to what is being recorded, how the collected data is used and stored, and whether it is being protected well [7–9, 11, 12]. Much of previous work has been centered around protecting sensitive information from being leaked to adversaries [7–9]. Other questions, such as what is being recorded and whether Echo is eavesdropping on users or not, remain unanswered. As an attempt to increase transparency, Amazon has made it possible for users to view, play, and delete the voice data transmitted to its server [10]. However, there is no guarantee that the audio made available by Amazon is all that collected. It is still possible that Echo streams unwanted audio data that happens before or after the command to its server. To our best knowledge, no previous work has tried to measure whether Echo is transmitting anything other than the desired command.

This is where our work comes in. We designed and implemented experiments that allowed us to infer the behavior of the Echo through network traffic analysis. In our experiment setup, we connected an Echo to a dedicated hotspot and used Wireshark to capture all the packets sent and received by Echo. To activate Echo, an audio command was needed. We prerecorded an audio file that contained a simple command ("Alexa, where is New York City?") that was about 2.5 s long. We also prefixed and suffixed the command with irrelevant conversations. This prerecorded audio gave us both repeatability and flexibility. First, by tuning when to start and stop, different segments of this audio could be played. Second, the same segment could be played over and over again, with the guarantee that the input stays the same. We then carried out a set of experiments with this kind of setup.

Our first contribution: we confirm that Echo usually only transmits audio that happened after the wake word. In this experiment, we used two segments from the prerecorded audio. The first audio segment (denoted as I_1) was about 2.5 s long and contained only the Echo command "Alexa, where is New York City?". The other audio segment (denoted as I_2) was about 11.5 s long, with a 9 s irrelevant conversation attached right before the Echo command. We played both I_1 and I_2 alternately overnight for over 150 times, giving us in total more than 300 samples. For each sample, after data cleaning, we calculated the total size of all outgoing traffic. We then computed the mean and standard deviation of the total size across all samples of I_1 and I_2, as shown in Table 1. We did not observe any significant difference in the total size of outgoing packets between I_1 and I_2. Our hypothesis was that if Echo was sending any audio that happened before the wake word, we should be able to observe different total sizes of outgoing traffic for I_1 and I_2, which we did not. Thus, we concluded that nothing before the wake word was streamed to the server.

That being said, we did observe a few outliers that sent significantly more data than expected. Since we were unable to decrypt the traffic, we were unsure if our conclusion would still hold for these corner cases.

Our second contribution: we characterized Echo's different behaviors when it succeeded in detecting the end of a command and when it failed to do so. In this experiment, we composed a set of input audios. We started by taking two segments from the prerecorded audio. One was the Echo command (denoted as I_3), and the other was a random conversation (denoted as I_4) that

Table 1: Mean, standard deviation (SD), and median of total size (in kB) of outbound traffic across all samples of I_1 and I_2

Audio Input	Mean	SD	Median
I_1	44.94	0.68	44.85
I_2	44.81	0.60	44.64

was about 7 s long. We then crafted a series of input audio I_5 to I_{12} by concatenating I_3, x seconds of pause, and I_4 together, where x varies from 0.1 to 0.8 s with step size 0.1 s. As a result, I_5 consists of I_3, plus a 0.1 s pause, plus I_4. Similarly I_{12} consists of I_3, plus a 0.8 s pause, plus I_4. We played I_5 through I_{12} alternately, as we did in the experiment mentioned above. By looking at the inbound and outbound traffic, we were able to observe the following behaviors:

First, we identified that a 0.7 s pause was needed after the command for Echo to successfully sense the end of a command. For each x second gap, we recorded whether the Echo successfully identified the end of the command (I_3) and provided a response while I_4 was still playing, or whether instead it waited until the end of I_4 to respond, trying to parse the entire input as a command. We saw that below 0.5 s, the Echo never successfully identified the end of the command, and it wasn't until above 0.7 s that the Echo always detected the end of the command.

Second, we verified that the Echo does not record any audio past the end of a command when such an end is properly detected. For I_{11} and I_{12}, where the pause was clearly long enough for the Echo to detect the end of the command, we observed similar amounts of outgoing traffic as we did for I_1 and I_2, implying that Echo did not send anything other the command.

Third, if the Echo fails to detect the end of the command, it will continue to record audio. For I_5 and I_6, where the pause was clearly not long enough for the Echo to detect the end of the command, we observed much larger outgoing traffic than we did for I_1 and I_2, implying that the Echo did send other audio data than just the command.

In sum, we performed network traffic measurement on the Echo that allowed us to infer the Echo's behavior and address concerns regarding it. We confirmed that the Echo in most cases did not record any audio that happened before the wake up word, and that it would not stream any audio after the end of a command if it was able to detect the end. In the case of failing to detect the end of a command, the Echo did send more audio data than just the command to its server.

2 BACKGROUND

2.1 Amazon Echo

Amazon Echo is a smart speaker developed by Amazon that is connected to the Internet [3]. It has an always-on microphone that starts recording automatically after hearing a certain wake word (Alexa, Echo, *etc.*). Voice commands following the wake word are streamed to Amazon's cloud-based intelligent command handling program, known as Alexa, for further processing. Alexa will then try to respond to users' commands. Amazon also keeps copies of the voice commands and responses, together known as response cards [10]. These response cards have been made available to users by Amazon so that they can be viewed, played, and deleted [1].

2.2 Wireshark

Wireshark [5] is a tool for network traffic monitoring. It can intercept and record all network traffic from a specific interface and convert that binary traffic into human-readable format. It has made it easy for humans to identify how much traffic is being sent and received, as well as who is the sender or receiver.

3 EXPERIMENT

We ran some experiments on Alexa and use Wireshark to see if Alexa recorded consumer's voice more than we suppose it does.

3.1 Monitoring Set Up

In order to catch the packets that were sent out by Alexa to the Amazon cloud server, we set up a WiFi router on a desktop server and connected the Echo to it. We run Wireshark [6] on the desktop server to capture all the packets to and from the Echo speaker.

Since we cannot directly examine the contents of the packets sent to and from the Echo, as they are all encrypted by TLS, we instead must examine coarser-grained information, like packet count, packet sizes, server IP addresses, and packet timing information. Since most of this kind of coarser-grained information is subject to variation based on network conditions, and the voice data itself is subject to variation based on environmental noise, we opt to repeat all of our experiments many times (10s to 100s) to allow us to draw statistical conclusions.

To ensure the repeatability of such experiments, we synthesize all our voice commands and play them to the Echo using the desktop server. We also use it record any response that the Echo may give, which allows us to gather all relevant data for our experiments on a single, centralized platform. Each experiment consists of some number of different voice commands that we iterate through throughout the experiment. We leave 30 s to a minute between each command to allow the Echo to reset, and after reaching the end of our set of commands, we loop back to the beginning. We perform each experiment overnight in the office, to remove any extra human noise and ensure that the Echo only responds to the commands we provide it. We interleave the commands that we are testing to make sure that any variation throughout the night due to conditions we cannot account for affect each of the commands equally.

To make analysis easier, we use the desktop server to inject markers into the Wireshark packet capture at critical moments. This allows the analysis code to read only the single Wireshark capture and still obtain all necessary information. In particular, the desktop server pings four different predefined locations (*e.g.*, 8.8.8.8) at four specific times: when the command starts playing, when the command stops playing, when Alexa's response begins, and when Alexa's response ends.

3.2 Analysis Techniques

After an overnight run of an experiment, we programmatically divide the Wireshark capture into experiment frames (individual command trials). We define these frames using the ping packets generated by the desktop server; the extent of each frame spans from the start of the voice command to the start of Alexa's response. This captures all the data sent by the Echo to the Amazon servers and the response from the Amazon servers back to the Echo, since we know that such a response must be completed by the time the Echo speaks that response.

The Echo communicates with many different servers for background tasks, and even a couple different servers as part of processing a command. As shown in Fig. 1, though, the Echo sends the majority of its data to a single IP address, namely the Amazon server handling the Alexa request. For each frame, we choose to

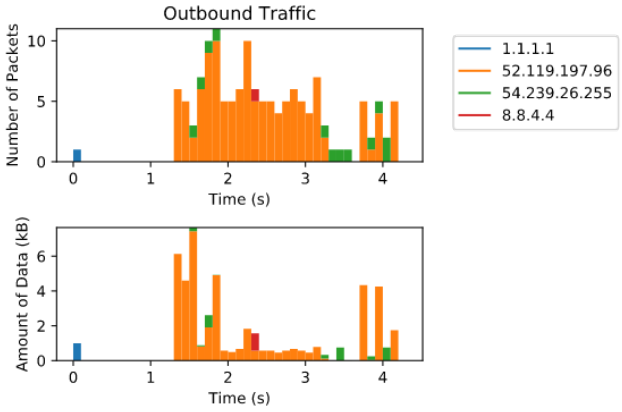

Figure 1: On top, time histogram of the number of packets sent from the Echo, sorted by destination IP address. On bottom, time histogram of total data sent from the Echo, sorted in the same way. The blue and red bars do not indicate real traffic, but rather mark the start and end of the command given to the Echo, as discussed in Section 3.1. The time histogram ends at the moment the Echo begins its reply. The orange bars are those of the Alexa server, in this case, as they constitute by far the most data.

only count the data going to this server, as it provides the most consistent metric to compare across. The precise identity of the Alexa server changes fairly frequently, so in the course of our analyses we used two different methods to identify the server in each frame: (1) we manually inspected many packet data time histograms (as in Fig. 1) to create a whitelist of Alexa serves, and (2) we simply chose the IP address in each frame that had the most data sent to it. Both methods gave very similar results, so we used the latter for most analyses, as it was more automated.

Some of the frames had much more data sent in them than others did, and inspection of their time histograms generally revealed that this was due to unusual traffic patters clearly not related to our commands (perhaps background updates or other such maintenance). These outliers needed to be filtered, as in some cases they were so high as to completely change the mean and standard variation of the entire data set. To filter such outliers, we decided to use the IQR (inter-quartile range) method to remove them. The IQR is defined as the difference of the 3rd quartile value and the 1st quartile value, and we filter any value that is more than 1.5*IQR higher than the 3rd quartile value or 1.5*IQR lower than the 1st quartile value. In other words, we keep a value x only if $Q1 - 1.5 * IQR < x < Q3 + 1.5 * IQR$. We chose this method because of its high robustness to large outliers.

We choose to count only the data sent in TLS packets, as the Echo encrypts all of its traffic, and so any other packets are only handshakes, DNS requests, or other such administrative packets. We are careful to count only the size of the actual TLS payload, and not any of the header bytes, so as to ensure the most accurate accounting possible.

4 MEASUREMENTS

4.1 "Prefix" Experiment

Amazon specifies in their document [2], "By default, Alexa-enabled devices only stream audio to the cloud if the wake word is detected (or Alexa is activated by pressing a button)." They claim they will not transmit any data before the wake up word. In order to test out this claim, we gather data as Alexa transmits voice to the server.

Therefore, in this part, we did two different kinds of experiments. First, we played an audio clip containing a prefix phrase as well as a command and another audio clip without a prefix phrase, but just command for Alexa. The prefix phrase is a random phrase that simulates a person talking in his daily life, who then asks Alexa a question. The whole prefix phrase is, "I think I am having some trouble hearing what people trying to say to me. I am not feeling particularly well now. I am also thinking of switching to a new job," and it played for 9 s. The command is: "Alexa, where is New York City?" and it played for 2.5 s. As a result, if Alexa did record some part of the prefix phrase, the total data transmitted for these two audio clips would be significantly different.

Our first experiment varied the length of the prefix phrase from 9 s to 0 s, which was then followed by the 2.5 s command. Our assumption is that, if Alexa truly does the transmission as Amazon states, the size of each transmission package should be the same, as there is only a short command at the end of the phrase. However, if Alexa's recording algorithm misbehaves at all, it might transmit some parts of the prefix phrase to the cloud, in which case the total data sent would vary with the prefix length.

Fig. 2 shows the result of this experiment. It clearly shows that there is no significant difference in the amount of data sent between any of the prefix lengths, as the standard deviation of any one of the points is bigger than the largest gap between any of the points. We were concerned that this experiment simply didn't have enough trials to show any possible correlations, as each command was only played about 15 times, so we ran another experiment, shown in Fig. 3, that only used two audio clips: one with the full 9 s prefix and one with no prefix. This allowed each command to be run several hundred times. The plot still clearly shows no significant difference between the two commands, so we conclude that the Echo does in fact only send audio captured after the wake-up word.

4.2 "Postfix" Experiment

We also want to measure whether Alexa will keep transmitting packages even if Alexa has noticed the whole command and started replying. The assumption here is that if Alexa has started replying, then Alexa has fully recognized the whole command and the Echo should not transmit anything else to the server. The experiment fits a real life situation where someone sends a command to Alexa and starts another conversation with others immediately. If Alexa transmits the rest or part of that conversation, it would definitely raise a privacy concern.

Here we use, "Alexa, where is New York City?" as a command sentence and add a 1 s postfix sentence. We gave an approximately 0.5 s gap between the command and postfix sentence to simulate the normal silent gap when people finish a sentence. Then we started playing three different audio clips: a) the command alone, b) the

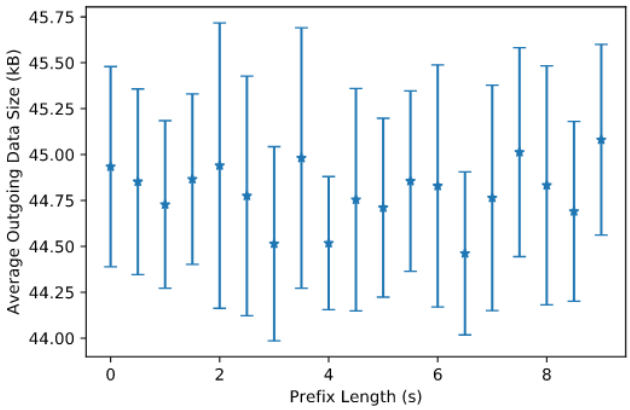

Figure 2: Plot of the total outgoing data size against the length of non-command speech played immediately before a command. The error bars extend to one standard deviation in either direction. Total outgoing data size is calculated as the sum of the TLS packet payload lengths going to the destination IP address with the greatest such sum.

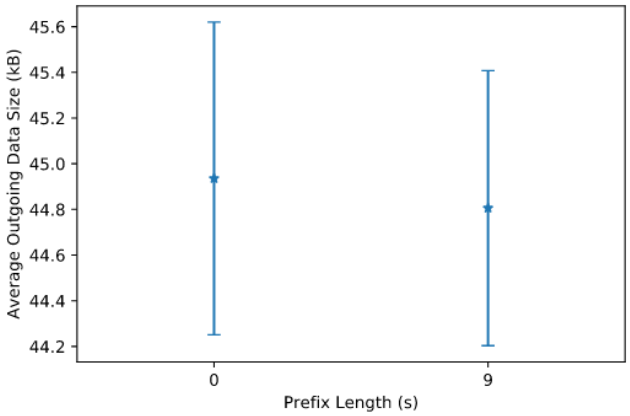

Figure 3: Plot of the total outgoing data size against the length of non-command speech played immediately before a command. The error bars extend to one standard deviation in either direction. Total outgoing data size is calculated as the sum of the TLS packet payload lengths going to the destination IP address with the greatest such sum.

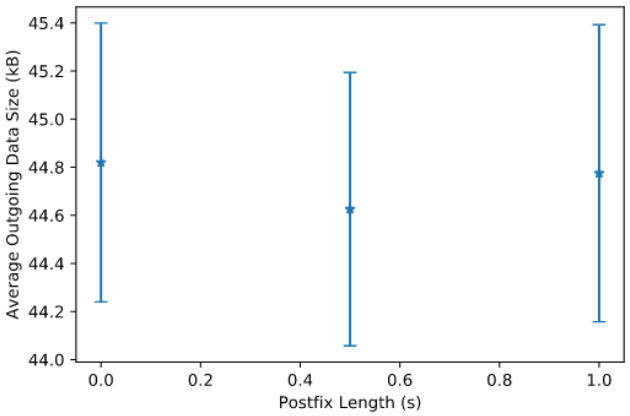

Figure 4: Plot of the total outgoing data size against the length of non-command speech played 0.5 s after a command. The error bars extend to one standard deviation in either direction. Total outgoing data size is calculated as the sum of the TLS packet payload lengths going to the destination IP address with the greatest such sum.

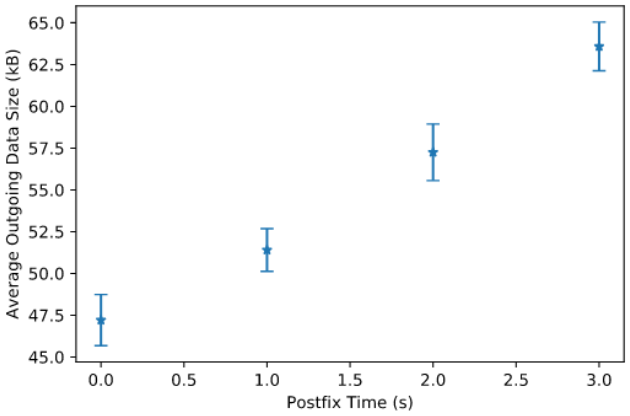

Figure 5: Plot of the total outgoing data size against the length of non-command speech played immediately after a command. The error bars extend to one standard deviation in either direction. Total outgoing data size is calculated as the sum of the TLS packet payload lengths going to the destination IP address with the greatest such sum.

command with a 0.5 s postfix sentence, and c) the command with a 1 s postfix sentence.

Fig. 4 shows that, again, there is no significant difference between the three trials that we run here. This confirms that the Echo does what we would like it to do and stops transmitting as soon as it has detected the end of the command.

We expect, however, that the 0.5 s silent gap between command and postfix sentence is necessary for Alexa to realize the command is over. W ran another experiment where we played the command with postfix sentence immediately (*i.e.*, we deleted the gap between postfix sentence and command.). We try to find out whether Alexa would automatically cut off the voice once it recognized that the following sentence was meaningless. We played the command with 0, 1, 2, and 3 s postfix sentences, which contain nonsense speech, to see whether Alexa would cut off the transmission itself.

Fig. 5 confirms our assumption that Alexa needs some kind of gap to recognize the end of a sentence. Without such a gap, the results clearly show that the total transmitted data grows directly with the length of the postfix. This indicates that the Echo is recording the entire command with the postfix and transmitting the entire thing to the Amazon server.

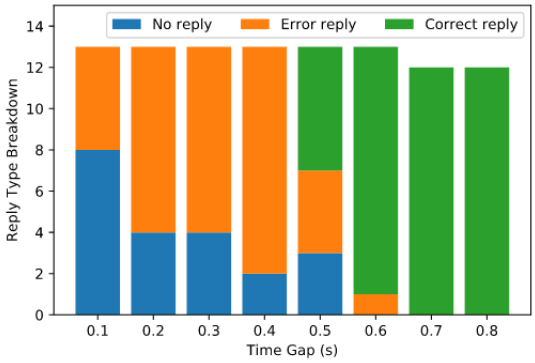

Figure 6: Bar plot showing how often Alexa replied correctly to a command, plotted for different gap lengths between the command and non-command speech.

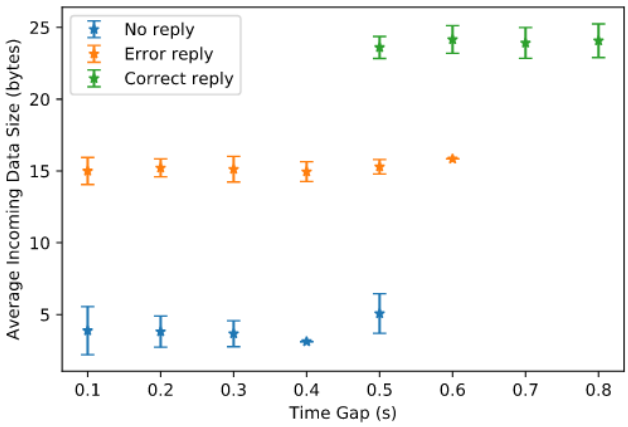

Figure 7: Plot of the total outgoing data size against the length of the gap before non-command speech is played after command. The error bars extend to one standard deviation in either direction. Total outgoing data size is calculated as the sum of the TLS packet payload lengths going to the destination IP address with the greatest such sum.

The previous two experiments raise the question of how long, exactly, does Alexa need to detect that a command is over and should therefore stop recording. We might want to know this so we know how long we need to pause before we start our conversation to let Alexa know that the command is over and can avoid Alexa transmitting our private conversation onto the Amazon server.

We attempt to determine this by playing the command with different silent gaps before playing the postfix sentence, to figure out when Alexa would reply correctly to our command. We added gaps that ranged from 0.1 to 0.8 s with 0.1 s steps.

Fig. 6 shows the results from this experiment. The blue indicates that Alexa does not respond to the sentence at all. The orange indicates that Alexa responds, but only after the entire postfix plays, and does not know what the command was supposed to be. The green indicates that Alexa noticed the end of the command correctly and immediately replied to that part instead of to the entire sentence. We can see that Alexa does not start correctly detecting the end of the command until at least a 0.5 s gap occurs, and the gap needs to be at least 0.7 s to ensure that Alexa will correctly detect the end, and not transmit any potentially private conversation.

To confirm our results, we recorded the total data that the Amazon server sent back to the Echo. We could see a significant difference among "non-response" packages, "error response" packages, and "correct response" packages, as shown in Fig. 7. The blue data points indicate the "non-response" experiment trials, the orange data points indicate the "error response" trials, and the the green ones indicate the "correct response" trials. We can clearly see the three levels of data, indicating separate responses to the voice command.

4.3 "Stop" Experiment

Following up on the above experiment, in this part, we study how Alexa detects the end of a command. We attempt to determine whether Alexa stops because of silence or becuase of some other detection method.

We tried three experiments here: a) We played background music while playing the command and kept playing the music for a while after the command ended to see if Alexa would stop transmitting packages and start replying to the command. This experiment can help us figure out if Alexa can detect the end of speech even without silence. b) We played the command with a background conversation simulating a situation in real life. For example, in a family, the father might ask Alexa something while the mother is talking to the child. The father's voice is louder, but the conversation between mother and child will keep going after the father finishes his command. Therefore, we would like to figure out whether Alexa can detect the end of the father's command. c) We start talking immediately after the wake up word and keep talking for a while without stopping to simulate a situation where in a party there is some conversation happening around Alexa with no natural breaks (due to multiple conversations overlapping). We want to figure out whether Alexa would stop recording automatically or will keep recording all of the sentences.

In the first experiment, where we played background music while talking to Alexa, we found that Alexa would stop transmitting packages to Amazon server and start replying, even though the music was still playing. This means Alexa can distinguish music from human voice.

However, when we played the command with some conversation in the background and kept the background conversation going after the command, we found that Alexa would keep recording if there were no significant silent gap in the conversation, even though the conversation and command were generated by different people. Therefore, Alexa is not able to detect different peoples' voices and stop automatically. This indicates that we should be wary of starting a new conversation while someone is using Alexa, as our conversation could be recorded by Alexa well past the end of the command to Alexa.

To simulating the situation that a few conversations are happening around Alexa and someone wakes up Alexa accidentally, we

played a meaningless phrase with the wake-up word at the beginning and then no pause in it. When we played an 18 s sentence, Alexa stopped at seemingly random points between 6–15 s into the sentence, before it actually ended. This seems to indicate that Alexa tries to protect privacy, once she determines the voice is meaningless.

5 CONCLUSION AND FUTURE WORK

We have presented some of the first measurements that seek to understand the behavior of the Echo in terms of what is being transmitted by the Echo. More specifically, we found that: 1) The Echo is not usually transmitting any audio that happens before it is activated; 2) In the case of correctly detecting the end of a command, Echo will not stream any conversation that occurs after the command; 3) A 0.7 s gap is needed for the Echo to correctly detect the end of a command. While there is a large amount of research on analyzing the EchoâĂŹs network traffic, ours is the first to use its network traffic to characterize its behavior quantitatively.

Despite having some inspiring results, we did notice some apparent limitations in this work and would leave them for future work. First, we only performed the experiment with one particular command. Thus, our findings do not necessarily generalize to scenarios where other Echo commands are used. Second, we observed a few cases in all of the experiments where the amount of data transmitted was unexpected. We were unable to analyze and explain what happened as we could not decrypt the data. Finally, there is the inherent weakness of our approach—inferring the Echo's behavior by measuring its traffic. Noise and other disturbances made it harder for us to draw any conclusion statistically.

REFERENCES

[1] 2010. GP. (2010). https://www.amazon.com/gp/help/customer/display.html?nodeId=201602040

[2] 2019. Alexa annoucement. (2019). https://www.amazon.com/gp/help/customer/display.html?nodeId=201602230&pop-up=1

[3] 2019. Amazon Echo. (Oct 2019). https://en.wikipedia.org/wiki/Amazon_Echo

[4] 2019. Amazon Echo (2nd generation) âĂŤ Alexa Speaker. https://www.amazon.com/Generation-improved-sound-powered-design/dp/B06XCM9LJ4?. (2019). (Accessed on 10/15/2019).

[5] 2019. Wireshark. (Dec 2019). https://en.wikipedia.org/wiki/Wireshark

[6] 2019. Wireshark. (2019). https://www.wireshark.org/

[7] Noah Apthorpe, Danny Yuxing Huang, Dillon Reisman, Arvind Narayanan, and Nick Feamster. 2019. Keeping the smart home private with smart (er) iot traffic shaping. *Proceedings on Privacy Enhancing Technologies* 2019, 3 (2019), 128–148.

[8] Noah Apthorpe, Dillon Reisman, and Nick Feamster. 2017. A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic. *arXiv preprint arXiv:1705.06805* (2017).

[9] Noah Apthorpe, Dillon Reisman, Srikanth Sundaresan, Arvind Narayanan, and Nick Feamster. 2017. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *arXiv preprint arXiv:1708.05044* (2017).

[10] Marcia Ford and William Palmer. 2019. Alexa, are you listening to me? An analysis of Alexa voice service network traffic. *Personal and Ubiquitous Computing* 23, 1 (2019), 67–79.

[11] Geoffrey Fowler. 2019. Perspective | Alexa has been eavesdropping on you this whole time. (May 2019). https://www.washingtonpost.com/technology/2019/05/06/alexa-has-been-eavesdropping-you-this-whole-time/

[12] Josephine Lau, Benjamin Zimmerman, and Florian Schaub. 2018. Alexa, are you listening?: Privacy perceptions, concerns and privacy-seeking behaviors with smart speakers. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 102.