Multi-Class Traffic Morphing for Encrypted VoIP Communication

W. Brad Moore, Henry Tan, Micah Sherr, and Marcus A. Maloof

Georgetown University
{wbm,ztan,msherr,maloof}@cs.georgetown.edu

Abstract. In a *re-identification attack*, an adversary analyzes the sizes of intercepted encrypted VoIP packets to infer characteristics of the underlying audio—for example, the language or individual phrases spoken on the encrypted VoIP call. *Traffic morphing* has been proposed as a general solution for defending against such attacks. In traffic morphing, the sender pads ciphertext to obfuscate the distribution of packet sizes, impairing the adversary's ability to accurately identify features of the plaintext.

This paper makes several contributions to traffic morphing defenses. First, we argue that existing traffic morphing techniques are ineffective against certain re-identification attacks since they (i) require a priori knowledge of what information the adversary is trying to learn about the plaintext (e.g., language, the identity of the speaker, the speaker's gender, etc.), and (ii) perform poorly with a large number of classes. Second, we introduce new algorithms for traffic morphing that are more generally applicable and do not depend on assumptions about the goals of the adversary. Finally, we evaluate our defenses against re-identification attacks, and show, using a large real-world corpus of spoken audio samples, that our techniques reduce the adversary's accuracy by 94% with low computational and bandwidth overhead.

1 Introduction

Over the last decade, the use of voice-over-IP services as an alternative to landlines and mobile phones has dramatically increased. For instance, Skype calls accounted for just 2.9% of the international call market in 2005 [19]; by 2012, that percentage increased by an order of magnitude to 34% [20]. Between 2011 and 2012, the number of concurrent users online nearly doubled from 27 million to 50 million [16].

Additionally, VoIP offers the ability to more easily secure the communication content using end-to-end (e2e) encryption – either as part of the communication protocol (cf. Skype [3]) or by layering established cryptographic protocols such as SSL/TLS (cf. WebRTC). The widespread adoption of encrypted VoIP services such as Skype implies a more secure communication infrastructure that is resistant to eavesdropping.

However, existing work has shown that even when strong encryption is applied, encrypted VoIP streams often leak significant information about the plaintext audio. To conserve bandwidth, most popular VoIP systems make use of variable bit-rate (VBR) encoders in which the amount of output data per time unit varies according to the complexity of the audio sample. Importantly, although VoIP systems may encrypt audio

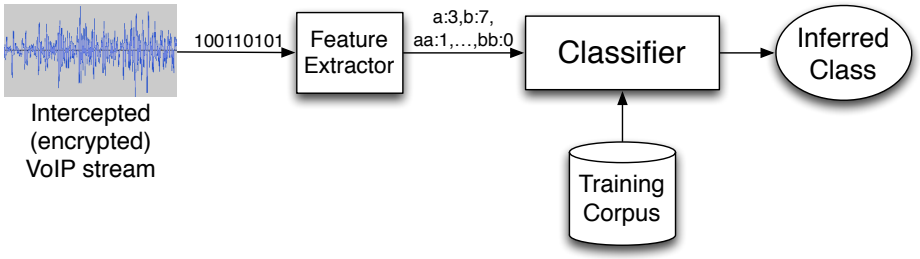

Fig. 1: Attack workflow. An adversary conducts a re-identification attack by extracting features from an intercepted, encrypted stream. In this example, each observed packet size is mapped to a symbol from the alphabet $\Sigma = \{a, b\}$. The feature extractor counts the number of occurrences of each symbol ("unigram") and adjacent pair of symbols ("bigram"). Using a corpus of labeled training data whose features have been similarly extracted, the attacker uses machine learning techniques to infer the class of the intercepted communication (e.g., the speaker is speaking German or is Groucho Marx).

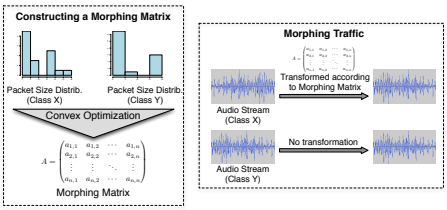

Fig. 2: Conceptual overview of the traffic morphing approach by Wright et al. [28].

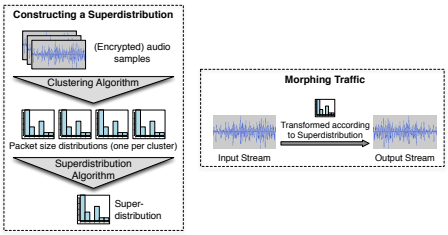

Fig. 3: Conceptual overview of Muffler.

packets, the systems' underlying use of VBR induces a side-channel through which an adversary may infer information about the plaintext by observing only the sizes of encrypted packets. In particular, Wright et al. showed that such observations are sufficient to accurately infer the language being spoken [26]. As shown in Figure 1, an adversary can extract features from the ciphertext based on packets sizes and use machine learning techniques to infer attributes of the underlying plaintext. Followup work by many of the same authors demonstrated that machine learning techniques could additionally identify speakers [13] and phrases [24, 27] with high accuracy. Throughout this paper, we use the term *class* to denote a group of audio samples that share a common attribute (e.g., speaker's gender, speaker's identity, or spoken language).

We present a novel blackbox approach that we call *Muffler* to defend against traffic analysis of encrypted VoIP streams. As with other blackbox defenses [28], we assume a closed-source VoIP client that sends encrypted packets, e.g., Skype. To maintain compatibility with existing applications, Muffler operates as an add-on security layer; we make no modifications to client software.

1.1 Strawman Defenses

An obvious and simple defense against VoIP re-identification attacks is to replace VBR codecs with CBR encoding. However, this strategy would require either degradation of call quality, or a significant increase to stream bandwidth. CBR encoding is not well-suited for networks with limited bandwidth (e.g., mobile data networks) – our aim is to develop defenses that incur low bandwidth overheads.

1.2 Background: Traffic Morphing

To defeat traffic analysis while maintaining high-quality audio, Wright et al. proposed a *traffic morphing* approach in which one class of traffic is transformed to match the statistical properties of another existing class [28]. Specifically, they selectively add padding to packets to obfuscate a stream's true distribution of packet sizes and make the stream appear indistinguishable from another distribution while minimizing the amount of padding necessary. Using a comparison function such as the χ^2 statistic and convex optimization, they find the distribution closest to a target distribution that is attainable by padding (see Figure 2, left). The result is a *morphing matrix* A where each value a_{ij} in the matrix represents the probability that an (encrypted) audio sample of size s_i is padded to s_j (see Figure 2, right). They show that for binary classification, their traffic morphing technique significantly degrades the accuracy of the classifier from 71% (without obfuscation) to 30%, while incurring a communication overhead of 15.4% [28].

In a blackbox design, packet padding can be achieved by tunneling the encrypted VoIP packets in another layer of encryption where padding may be added.

The receiver decrypts this layer and discards the padding to obtain the original encrypted VoIP stream. Importantly, while the sender can pad packets, packet sizes cannot be decreased since the VoIP client functions as a blackbox. That is, if $s_j < s_i$, then $a_{ij} = 0$ in the morphing matrix.

1.3 A New Approach to Blackbox Traffic Morphing

This paper proposes a new traffic morphing technique that we call Muffler. In contrast to existing work in which one distribution is morphed to a another 'target' distribution, we construct a new synthetic distribution to which all input audio streams are morphed. Also, as discussed in the next section, a limitation of existing techniques is that they assume that the sender knows the adversary's intent (e.g., to determine if the speaker is speaking English or German). With Muffler, we adopt a stronger threat model and assume that the sender does not know the adversary's classification task (language, speaker, gender re-identification, etc.).

Figure 3 presents a high-level overview of Muffler. Given a background corpus of encrypted audio (either labeled or unlabeled), Muffler uses clustering techniques to form groups of samples, where each group could potentially be a classification used by an adversary in a re-identification attack. For example, a cluster of samples could correspond to spoken Arabic, female speakers, et cetera. Using these clusters, Muffler creates a "superdistribution" of packet sizes to which *all* discovered clusters may be mapped.

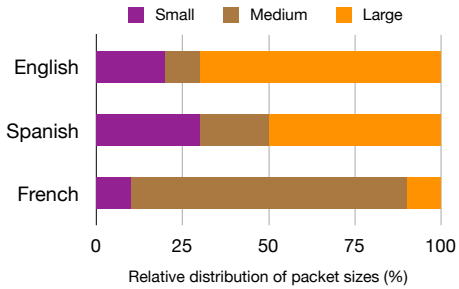

Fig. 4: A hypothetical example of packet size distributions for spoken English, French, and Spanish. Although the distribution of Spanish may be morphed (by padding) to appear as the distribution of English, the reverse is not possible. None of the three distributions is a viable target to which the other two distributions may be mapped.

Using a large suite of classifiers, we demonstrate that Muffler effectively thwarts traffic analysis of encrypted VoIP streams with both low computation and bandwidth overheads.

2 Improved Traffic Morphing

The state-of-the-art defense against re-identification attacks is the traffic morphing approach introduced by Wright et al. [28]. (We survey other related literature in Section 8.) In this section, we highlight some of the advantages of Muffler over this existing work.

2.1 Lightweight Traffic Morphing

The approach taken by Wright et al. uses convex optimization to find the best-matching distribution between two audio streams. Calculating the optimal stream transformation requires over an hour on their tested audio samples [28]. Muffler avoids expensive convex optimizations and is therefore able to *dynamically* adapt to the input signal: we show that our processes are sufficiently lightweight to adjust the transformation mappings in real-time.

2.2 Finding a Morphable Distribution

As described above, we consider blackbox defenses, where the traffic morphing approach may only increase packet sizes. Wright et al. study both whitebox and blackbox solutions, where packet sizes may be decreased in the former case, e.g., by temporarily using a lower bitrate. For whitebox systems, their traffic morphing scheme optimally morphs one distribution into another. For clarity, we will refer to the two distributions throughout this paper as the "source" and "target", respectively.

For blackbox settings, where the only permitted operation is padding packets, it may be impossible to transform an existing distribution into another existing distribution. Consider the example distributions of packet sizes for spoken English, French, and Spanish depicted in Figure 4. Morphing the Spanish distribution to appear as the English distribution is straightforward: a portion of Spanish small packets are padded to appear as medium packets, and a greater portion of Spanish medium packets are padded to appear as large packets. However, given the distributions shown in the figure, the converse is not possible: an English speaker's traffic cannot be morphed into a

Spanish speaker's, and in fact none of these three distributions can be morphed to from both of the other two.

The key benefit of Muffler is that it calculates, based on a set of speakers' streams, the "superdistribution" *with minimal bandwidth cost* that may serve as the target distribution for any of the source distributions. Once the superdistribution is established, Muffler then uses lightweight traffic morphing techniques to map streams to the superdistribution.

2.3 Automated Class Detection

The traffic morphing technique introduced by Wright et al. requires labeled training data (e.g., audio samples that are marked as containing spoken English, French, or Spanish). To avoid this, we apply unsupervised clustering techniques to an unlabeled corpus of audio samples containing representative samples for the classes that an adversary may attempt to re-identify. An advantage of our unsupervised learning technique is that the classes need not be explicitly labeled in the corpus. Hence a large and diverse corpus of audio samples may be sufficient to construct a superdistribution that captures a large range of possible classifications.

In Section 4, we show that (1) simple clustering techniques are sufficient to detect classes, and (2) the performance of Muffler when clustering is used to detect classes is approximately equivalent to cases where the classes are explicitly specified.

3 System and Attacker Models

We consider two parties communicating via a stream of encrypted VoIP packets. Each packet represents a fixed-time audio sample encoded using a VBR codec. We denote the stream of audio samples recorded by the sender as an ordered list $S = \langle s_1, \ldots, s_m \rangle$. Let $v(s_i)$ be the output of sample s_i after encoding with the VBR codec and $E(v(s_i))$ be the encryption of that encoded output. Let $|E(v(s_i))|$ be the length (in bits) of that ciphertext. We define the alphabet Σ as the set of possible lengths of encrypted audio samples produced by the codec; i.e., $\cup_{s_i \in S}\{|E(v(s_i))|\} \subseteq \Sigma$, with equality usually being the case for spoken audio that is longer than a few seconds. Without loss of generality, we consider the symbols (packet sizes) in Σ to be ordered by size; that is, we set $\Sigma = \langle z_1, \ldots, z_n \rangle$ such that $z_i < z_j$ iff $i < j$. Finally, we assume that $\forall s_i \in S$, $|E(v(s_i))| - |v(s_i)| = c$, where $c \geq 0$ is a small constant. This latter assumption is necessary to allow an adversary to determine the size of the unencrypted audio sample $|v(s_i)|$ without knowledge of the sample or the decryption key. Or, equivalently, we assume that the audio samples have not been padded.

We model a passive adversary who intercepts all encrypted VoIP packets in the order in which they were sent. The adversary does not have access to the plaintext. Let $L = \langle l_1, \ldots, l_m \rangle$ be an ordered list of the lengths of ciphertexts for the stream S. That is, $l_i = |E(v(s_i))|, l_i \in \Sigma$. We note that the sequence L induces a distribution of packet lengths. The adversary's goal is to use the side-channel L to infer information about S.

We consider *classes* of speakers where speakers that share a particular attribute (e.g., gender) belong to the same class. Let $\mathcal{A} = \langle a_1, \ldots, a_q \rangle$ be the set of classes that

are of interest to the adversary. For a given sample S, we denote the correct class as $\overline{a}_s$. As with existing work, we assume that an audio stream has exactly one class. By assumption, $\overline{a}_s \in \mathcal{A}$ and, to avoid the trivial case, $|\mathcal{A}| > 1$.

We also conservatively assume that the adversary has access to a corpus Γ of unencrypted audio samples such that (i) $S \notin \Gamma$, (ii) for all $S' \in \Gamma, \overline{a}_{S'} \in \mathcal{A}$, (iii) for all $S' \in \Gamma$, $\overline{a}_{S'}$ is known to the adversary (i.e., the corpus is labeled with the correct classes), and (iv) the adversary may compute the lengths of ciphertexts $\langle l'_1, \ldots, l'_m \rangle$ produced by encoding and encrypting the audio of each sample in Γ. The first requirement ensures that the intercepted stream does not already appear in the corpus, while the second conservatively assumes that each sample in the corpus has a class in $\mathcal{A}$. Finally, for any audio stream S whose encoded ciphertext may be intercepted by the adversary, we assume that there are samples in Γ that belong to the class $\overline{a}_s$. We say that such a corpus provides *coverage* of the class $\overline{a}_s$.

Given L and Γ, the adversary's goal is to correctly infer $\overline{a}_s$—that is, to re-identify the audio's class. The goal of Muffler is to make the adversary's probability of correctly guessing $\overline{a}_s$ similar to the probability of guessing correctly without L.

As discussed above and visualized in Figure 1, the adversary may frame the re-identification task as a machine learning problem. For example, the approach by Wright et al. forms n-grams (overlapping segments of n-length sequences) over L, and uses a count of each n-gram as a feature for a machine learning classifier [26]. Using a background corpus to train the classifier, Wright et al. show that the adversary can reliably predict $\overline{a}_s$ when no obfuscation is applied. In Section 6 and in Appendix A, we formalize our security properties under the assumption that the adversary uses n-grams as features, noting that this approach is used by all re-identification attacks of which we are aware [24, 26–28].

With Muffler, we assume the speaker has access to a corpus of audio samples, Γ', that he may use to form a superdistribution to which traffic may be morphed (see Figure 3). As with Γ, we assume that Γ' provides coverage, i.e., there are samples in Γ' of the same class as the speaker's audio streams. Unlike Γ, we do not require that the samples in Γ' be labeled with their correct classes.

We envision that Γ' could be bundled with the Muffler software or obtained by the user. We note that acquiring a large corpora of speech is not particularly difficult: we use the public domain Librivox [15] collection of audio books; George Mason University maintains a set of more than 1,800 speech samples that cover a large range of languages and accents [1]; the University of Pennsylvania's Linguistic Data Consortium hosts hundreds of language corpora [2].

In this paper, we let $\Gamma' = \Gamma$. This is a conservative assumption, as it allows the adversary to train on the exact data used by the sender to form its superdistribution. (In cases where Muffler is applied, the adversary is allowed to train on the modified packets.)

Importantly, we note that a non-goal of our system is to provide *deniability*: Muffler does not attempt to conceal its use. Since Muffler morphs traffic to a superdistribution that may not resemble any non-obfuscated distribution, an adversary could use similar classification techniques to detect it. Our goal is to provide VoIP communication that resists re-identification attacks.

Algorithm 1 Given an array of distributions from a training corpus, calculate the superdistribution to which each input distribution may be mapped

```
1: proc calcSuperdistribution(arrayOfDistributions)
2: packetSizes ← ⟨z_n, ..., z_2, z_1⟩, n ← |packetSizes|
   {iterate through packet sizes, starting with the largest}
3: for all p_1 in packetSizes do
4:     Max ← largest frequency of packetsize p_1
       {given the maximum, create a superdistribution to which all other distributions may be morphed:}
5:     for all dist in arrayOfDistributions do
6:         deficit ← Max
7:         for all p_2 in packetSizes[packetSizes.index(p_1):n] do
8:             deficit←deficit−dist[p_2]
9:             if deficit < 0 then
10:                dist[p_2]← −1×deficit
11:                break
12:            end if
13:            if p_1 == p_2 then
14:                dist[p_2]← −1× Max
15:            else
16:                dist[p_2]← 0
17:            end if
18:        end for
19:    end for
20: end for
21: return ArrayOfDistributions[0]
   {After the last iteration, each distribution in arrayOfDistributions will be identical, and equal to the smallest
   possible (bandwidth-wise) distribution to which any of the distributions could be transformed.}
```

4 Forming the Superdistribution

In order to reduce the ability of an adversary to reliably determine any attributes of an audio stream, we aim to shape the distribution of the packet sizes within the streams such that classification (re-identification) is as difficult as possible. One method to make streams indistinguishable would be to pad each packet to the maximum size, which, while effective in preventing classification of speakers, negates the bandwidth savings achieved by using VBR in the first place. Another method is to attempt to pad a stream's packets in order to 'morph' the packet distribution to resemble that of other known streams. As discussed above, such an approach was explored by Wright et al. [28], and Muffler can also be categorized as a traffic morphing system. However, the approach by Wright et al. is not well-suited for disguising multiple classes, since not all streams are easily morphable to all other streams.

Muffler considers the distributions of all speakers in a training corpus, and then calculates the least bandwidth-intensive distribution to which *all* speakers in the corpus could be padded to. Specifically, letting $L_s = \langle l_{1_s}, \ldots, l_{m_s} \rangle$ be an ascending list of the m different posssible lengths of ciphertexts for a stream s: Muffler calculates superdistribution L_{super} such that for all $1 \leq n \leq m$, $\sum(l_{n_{super}} + \cdots + l_{m_{super}}) = \max_s(\sum(l_{n_s} + \cdots + l_{m_s}))$ over all streams s in the corpus.

The process used to calculate this superdistribution is presented as Algorithm 1. The algorithm is given an array of distributions such as that visualized in Figure 4. Each of the three bars in the figure reflects a different speaker class, the distinction between which our superdistribution will seek to eliminate. Note that Algorithm 1 creates a superdistribution from unigrams. (Section 4.1 presents the algorithm for n-grams.)

The algorithm works as follows: in line 4, the algorithm finds, amongst the input distributions, the largest count for the largest packet-size (z_n). In our example, the largest proportion of z_3 packets (where z_3 is the largest packet size) occurs in Spanish, and this maximum value is 30%.

Lines 5-20 describe the formation of the superdistribution. Conceptually, the superdistribution considers the relative frequencies of the packet sizes, in order of decreasing packet size. The superdistribution uses the largest relative frequency amongst the input distributions.

4.1 n-gram Superdistributions

Algorithm 1 calculates a superdistribution based on the distributions of packet sizes in a set of streams. However, morphing only unigrams may be insufficient if the adversary is using n-grams to classify streams. (In Section 7, we evaluate the effectiveness of a unigram-based superdistribution against an adversary who uses trigrams.)

To defend against n-gram adversaries, we construct multiple superdistributions. Muffler computes a superdistribution for each unique sequence of $n-1$ packet lengths. That is, if Muffler is considering trigrams and a packet length z_q in a sample is proceded by packet sizes z_a and z_b, then Muffler increments the counter for z_q in the distribution corresponding to the sequence $\langle z_a, z_b \rangle$. There will be $|\Sigma|^{n-1}$ such superdistributions.

Muffler uses this set of superdistributions, once built, to dynamically morph packets. Given the $(n-1)$ packets that were most recently output, Muffler uses the corresponding superdistribution (i.e., the one that matches the $(n-1)$-length sequence) to determine how the next packet should be morphed. The morphing operation is explained in more detail in Section 5.

4.2 Dynamic Clustering

Algorithm 1 takes as input an array of distributions of packet sizes, where each distribution within the array corresponds to a class (e.g., Spanish, English, and French, in the case of language re-identification). However, it may be the case that the classes are not known a priori—either because the corpus is unlabeled or the sender does not know the type of re-identification that the adversary will attempt (e.g., language vs. speaker re-identification). We expect that this latter case will be the norm in most deployment scenarios.

In light of this, we explored an alternate method of superdistribution generation in which the algorithm, given the set of streams as a whole, first creates its own classifications of the streams using an unsupervised clustering algorithm. In what follows, we will refer to this data preprocessing step as *dynamic clustering*.

We find that k-means clustering is sufficient to automatically generate the input distributions for Algorithm 1, given an unlabeled collection of audio samples. In Section 7, we show that Muffler is similarly able to mitigate re-identification attacks when the speaker's training corpus is (i) labeled or (ii) unlabeled and k-means clustering is applied. We discuss finding an appropriate value of k in Section 9.

Algorithm 2 Given a calculated superdistribution array, pad the packets of an input stream as necessary to map its distribution to the superdistribution

```
1: proc morphStream(packetInStream, targetDistro, numPossibleSizes, gramSize)
2: currentDistro ← an array of numPossibleSizes^gramSize empty distribution arrays
3: precedingNPackets← an empty queue of packet sizes
4: maxSizePacket ← a maximally-sized packet
5: for all x in range(0,gramSize) do
6:     currentPacket ←packetInStream.dequeue()
7:     packetOutStream.enqueue(maxSizePacket)
8:     precedingNPackets.enqueue(currentPacket.size())
9: end for
10: while currentPacket ← packetInStream.dequeue() do
11:     distributionDisparities ← an array noting the current distribution's disparity from the target distribution.
12:     for all possibleSize in range(currentPacket.size(),maxPossibleSize) do
13:         if currentDistro[precedingNPackets][possibleSize]/currentDistro[precedingNPackets][totalPackets] <
          targetDistro[precedingNPackets][possibleSize] then
14:             probabilitiesOfChoosing[possibleSize] ← (targetDistro[precedingNPackets][possibleSize] - cur-
              rentDistro[precedingNPackets][possibleSize]) x maxPossibleSize / (1 + possibleSize)
15:         else
16:             probabilitiesOfChoosing[possibleSize] ← 0
17:         end if
18:     end for
19:     chosenPacketSize ← WEIGHTEDCHOOSER(distributionDisparities)
20:     currentDistro[precedingNPackets][chosenPacketSize]++
21:     currentDistro[precedingNPackets][totalPackets]++ //totalPackets being the sum of all packet sizes
22:     precedingNPackets.enqueue(chosenPacketSize)
23:     precedingNPackets.dequeue()
24:     padAndSendPacket(currentPacket,chosenPacketSize)
25: end while
```

5 Mapping to the Superdistribution

Algorithm 2 describes how Muffler transforms an input stream to resemble a predetermined superdistribution. For clarity, we focus on the particular case in which the sender wishes to morph his traffic at the level of trigrams; we note that the algorithm works for any size n-gram.

In lines 5-8, we pad the initial $n-1$ packets to the maximum packet size. (Since packets usually convey 20ms of audio, this initial maximal padding is quickly amortized away.)

After this initial special case, the algorithm proceeds as follows: based on the previous $n-1$ outputted packet sizes (i.e., the packets that were transmitted after being morphed), we compare the target distribution of what should come next to the actual distribution of what has followed these two packet sizes in the current, obfuscated, output stream so far. In lines 13-17, the algorithm assigns probabilities to the possible choices for the packet size to output, based on which sizes are most underrepresented. These probabilities are skewed slightly toward smaller packets in line 14. On line 19, we use the WeightedChooser subroutine, which chooses a random packet size from those with disparities (i.e., distances from the superdistribution) greater than zero, weighted by the value of the disparity; or, if there are no such probabilities greater than zero, it returns the largest packet size. In lines 20-22, the current distribution is updated with the packet size we have chosen, and the precedingNPackets window is shifted forward to include this packet. In line 23, the current packet is sent, after being padded to the chosen packet size.

6 Security Analysis

Theorem 1. *Our scheme is IND-CGA (Indistinguishability against Chosen Generator Attack) secure.*

The IND-CGA game allows an adversary A to select a pair of *generators* g_0 and g_1, where generators are algorithmic models of speakers. Specifically, a generator outputs packet streams which share characteristics similar to those that would be produced by a particular speaker. From the pair of generators provided by the adversary, one such generator is chosen at random, with the choice being invisible to A. The randomly chosen generator is then used to produce a packet sequence, which is then morphed (using Muffler) and returned to A. Given g_0, g_1, the Muffler algorithms, and the morphed stream, the adversary's goal is to decide whether the randomly chosen generator was g_0 or g_1. Intuitively, if the adversary cannot make this determination *for any set of generators* g_0 *and* g_1, then the morphing provides a form of indistinguishability, which is exactly the goal of Muffler.

We remark that in the standard indistinguishability under chosen-plaintext attack (IND-CPA) game used to evaluate cryptosystems, the adversary there is allowed to choose arbitrary packet streams, as opposed to generators. As such, any reasonable blackbox morphing technique must morph all packets to the maximum size to be secure under IND-CPA, since the adversary could choose a sequence of all smallest-size packets and a sequence of all largest-size packets as his inputs. The adversary could then trivially identify that the smallest-size sequence was the one randomly chosen if the morphed sequence contains a single packet that is not the largest size. This applies to less extreme packet streams. As such, in our IND-CGA game, we restrict the adversary to choosing randomized generators whose output reflect real speech distributions.

In our analysis, we assume that the adversary performs classification by using n-grams as features. As explained in more detail in Appendix A, we believe that this is a realistic assumption, at least given currently known re-identification attacks, all of which (to the best of our knowledge) perform classification using n-grams (cf. [24, 26–28]). In Appendix A, we show that Muffler achieves IND-CGA under assumptions that existing work and our empirical results indicate hold true in practice.

7 Evaluation

Dataset We gather public-domain audio from Librivox [15], a collection of literature read aloud by volunteers. This source of data is especially good for our purposes, as the variance in background noise as well as the quality and frequency response of the microphones being used are all factors that affect the ability of a codec to compress the audio stream; this makes traffic morphing more difficult, since the streams are more easily distinguishable than if they were all recorded in a controlled environment with identical equipment. From the Librivox dataset, we extract 100 samples of 200 seconds of audio from each of 158 different speakers, totaling nearly 878 hours of audio.

We encode the audio samples from the Librivox dataset using the Silk codec [21] (the same codec used by Skype until late last year, and the basis for the current codec).

The output of this encoding step is a series of discrete audio packets. Using Silk's default parameters, there are eight possible sizes for the encoded audio; i.e., $|\Sigma| = 8$. Since we assume that the adversary is not able to decipher the traffic, we consider only the sizes, and not the content, of these packets.

Methodology In order to measure the efficacy of Muffler, we compare an adversary's ability to classify VoIP streams without any obfuscation beyond basic encryption, to an adversary's ability to classify streams that have been morphed with Muffler. The adversary's goal is to identify the speaker of an intercepted stream, from amongst the 158 speakers in our dataset.

Each sample in the Librivox dataset is an ordered sequence of packet sizes, $L = \langle l_1, \ldots, l_m \rangle$. From this sequence, we count the occurrences of unigrams, bigrams, and trigrams, where a unigram is a symbol, a bigram is a subsequence of two contiguous symbols, etc. The counts for each unigram, bigram, etc. are used as features for a machine learning classifier. For supervised learning, each sample is labeled with its correct class (as specified in the Librivox data). In this paper, we present results for adversaries using (i) unigrams and (ii) unigrams, bigrams, and trigrams.

The adversary uses a battery of classifiers: three variations of k-Nearest-Neighbor and Naïve Bayes, the J48 decision tree algorithm (based on C4.5), and a support vector machine (SVM) [25]. For the adversary who examines only unigrams, the training corpus contains only unigram counts; the stronger adversary has counts for unigrams, bigrams, and trigrams as training features.

To evaluate the efficacy of Muffler to mitigate re-identification attacks, we compare the adversary's ability to correctly classify streams with Muffler and without any attempted traffic morphing. For each configuration, we report the *mean classification accuracy* amongst all the machine learning classifiers and the *worst case accuracy*—i.e., the classification accuracy of the best performing classifier (and the worst-case accuracy from the perspective of the communicants).

For the results presented below, we use five-fold cross-validation. We conservatively assume that the adversary has access to the same corpus used by Muffler to form the superdistribution; that is, $\Gamma = \Gamma'$. However, the adversary always has access to a labeled training corpus; when Muffler uses dynamic clustering, we assume that the speaker does not know the class that interests the adversary (language, speaker identity, etc.) and consequently remove the labels from Γ'. For dynamic clustering, we use k-means clustering with $k = 32$.

For all cases where Muffler has been applied, the adversary allowed to train on the morphed versions of the packet streams.

7.1 Baseline Classification

Without any obfuscation (other than the encryption of packets), each of these classifiers is extremely adept at classifying speakers. The adversary's unigram classifiers average 26.3% accuracy in identifying the speaker, among 158 possible speakers, with the best classifier being able to correctly classify the speaker 28.1% of the time. Trigram classifiers average 43.3% accuracy in identifying the speaker, with a worst case accuracy of 72.4%, provided by SVM.

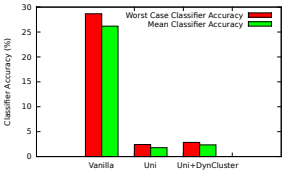

Fig. 5: Unigram-based adversary accuracy, with no morphing (Vanilla), Muffler using a labeled training corpus (Uni), and Muffler with k-means clustering ($k = 32$) applied to an unlabeled training corpus (Uni+DynCluster).

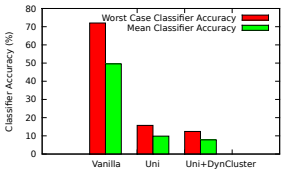

Fig. 6: Accuracy of re-identification when adversary considers the frequencies of trigrams. *Left:* No morphing. *Center:* Labeled training corpus, considers only unigrams. *Right:* Unlabeled corpus, considers only unigrams.

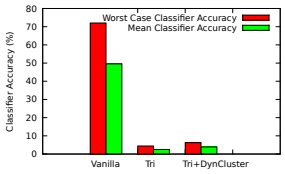

Fig. 7: Accuracy of re-identification when adversary considers the frequencies of trigrams. *Left:* No morphing. *Center:* Labeled corpus, and considers trigrams. *Right:* Unlabeled corpus, and considers trigrams.

7.2 Obfuscation against Unigram Classifiers

Using our method for unigram distribution obfuscation, we are able to significantly reduce the average and worst case accuracies of the unigram classifier battery: Figure 5 illustrates the accuracy of the classifiers before and after Muffler has been applied. Applying our unigram obfuscation technique reduces the average accuracy of the classifiers from 26.2% to 1.8%, and the worst case accuracy from 28.6% to 2.4% when Muffler has access to a labeled training corpus.

The bars marked "Uni+DynCluster" in Figure 5 show Muffler's accuracy when provided an unlabeled training corpus. (The corpus used by the adversary to train his classifiers remains labeled.) Here, k-means clustering is used on the entire set of audio streams in the training corpus Γ', and the resulting clusters are used as speaker classes by Muffler. The superdistribution is calculated by combining these 32 distributions.

The high comparative efficacy of our algorithm when using dynamic clustering is important to note. The similar performance of our algorithm when using dynamic clustering versus using a priori knowledge of class divisions means that deployment of Muffler would have very few technical hurdles: concealing speakers within a network could be achieved by simply placing Muffler at the edge of that network.

7.3 Obfuscation against Trigram Classifiers

Unigram-based traffic morphing is less effective when the adversary classifies streams based on longer n-grams.[1] Figure 6 shows the accuracy of classification based on trigrams on our audio streams. These "trigram classifiers" achieve very high accuracy on unobfuscated streams, with a worst-case accuracy of 72%, and an average-case accuracy of 49.6%, as can be observed from the first pair of bars. Muffler significantly degrades the accuracy of re-identification, providing mean and worst-case classification rates of 9.8% and 15.8% respectively.

[1] There is, of course, decreasing returns when n is large. As n increases, there are more unique n-length sequences and each are less likely to occur in the test data; hence, they provide less predictive value.

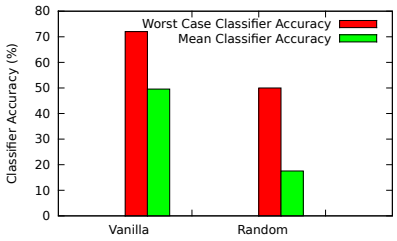

Fig. 8: Accuracy of re-identification when the adversary considers the frequencies of unigrams, bigrams, and trigrams for streams without obfuscation *(left)* and streams with random padding *(right)*.

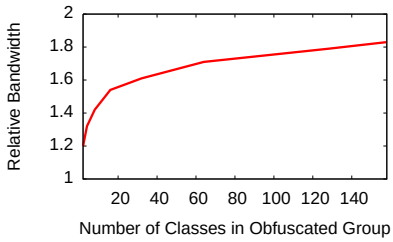

Fig. 9: Relative bandwidth overhead (w.r.t. unmorphed streams) of Muffler when considering uni-, bi-, and trigrams to build the superdistribution, as a function of the number of classes in the Librivox training data.

When we use our trigram-based superdistribution, the adversary's classifier accuracy drops even more. As seen in the middle bars of Figure 7, the worst- and average-case accuracies drop to 4.6% and 2.8%, respectively. This represents a reduction in accuracy of 94%, for the worst-case, when compared to unmorphed traffic.

7.4 Random Padding

We wish to show that the decreased accuracy of re-identification is not merely due to padding the streams away from their original form, but rather is attributable to morphing traffic to the superdistribution. We implement a simpler traffic morphing algorithm that randomly pads each packet in a stream by an amount adjusted such that the bandwidth cost of this random padding was similar to that of Muffler. As expected, while the padding did slightly decrease the adversary's ability to classify speakers, its efficacy at this task was far below that of our trigram superdistribution obfuscation technique, as shown in Figure 8. While the average classifier accuracy dropped to 17.5%, worst-case accuracy stood at 50.0%. As mentioned in Section 7.3, the equivalent worst-case accuracy for Muffler is 4.4%.

7.5 The Cost of Privacy

Figure 9 shows the relative cost of Muffler using a unigram superdistribution on the Librivox dataset, compared to the unmodified stream's bandwidth, for various numbers of speakers (classes) from which superdistributions are created. Because the cost of creating a superdistribution from a set of speakers depends on which speakers are included in that set, for each set size, we take a random sample of 16 possible combinations, and average the results to arrive at the data in the figure. Creating a superdistribution between two speakers in the set has a 20% bandwidth cost, while a superdistribution from 128 speakers incurs a 79% increase in bandwidth, on average. By comparison, the cost of full padding to the largest packet size (roughly analogous to using a constant bitrate audio codec) is a 171% increase over the original stream's size.

7.6 CPU Overhead

In comparison to existing traffic morphing techniques, Muffler avoids expensive operations and has a low CPU overhead. To build the superdistribution and morph the entire 878-hour corpus of audio from Librivox takes Muffler just under 30 minutes on a 3.1GHz Xeon E31220 with 8GB of DDR3 memory. This factor of 1,765 between the CPU time and amount of audio processed in that time means that it is entirely possible to have a Muffler implementation that dynamically updates the superdistribution being mapped to regularly, even while obfuscating several audio streams at once.

8 Related Work

Website fingerprinting. Much of the early work in packet- and stream-based traffic analysis focused on identifying the webpages conveyed in intercepted HTTPS streams. Sun et al. showed that the web page being visited by a user over an SSL-encrypted connection can often be identified based solely on the sizes of the objects being accessed. They additionally showed that this attack was resilient against padding object sizes as an obfuscation technique [18]. Later, Hintz introduced website fingerprinting techniques that infer the identity of a requested website by examining the size of an observed HTTPS stream [10]. In addition to inferring content, website fingerprinting has also been proposed as a method to defeat anonymity systems (most notably, Tor [6]) by identifying the webpages that have been requested by an observed client [5, 9, 22, 23]. Kadianakis [12] has suggested applying a variant of Wright et al.'s traffic morphing technique [28] to protect Tor against fingerprinting attacks.

Voice-over-IP. A series of papers including the work of Wright et al. discussed earlier have examined traffic analysis as a means to infer attributes about the audio signal embedded in an encrypted VoIP stream, and explored morphing techniques to disguise one class of speaker as another [26, 28]. However, when there are more than two possible classifications, they do not explore *which* distribution should be chosen as the target distribution. Subsequent work by many of the same authors showed that particular phrases can be identified by observing only the sizes of encrypted packets [24, 27]. Similarly, Khan et al. demonstrated that the adversary can identify the speaker of a conversation given a set of potential speakers, a corpus of their speech, and the encrypted VoIP stream [13].

Defenses. Developing defenses against traffic analysis is a growing area of research. Liberatore and Levine proposed padding packets up to the network MTU as a defense against web fingerprinting attacks [14]. However, recent work by Dyer et al. showed that such a strategy is ineffective against an adversary who employs a Naïve Bayes or a support vector machine classifier [7]. Folga et al. [8] explored the use of polymorphic blending to evade detection by intrustion detection systems. Their polymorphic blending approach included altering payload characteristics such as the byte frequency to resemble normal traffic. Iacovazzi and Baiocchi explored finding optimally efficient (with respect to bandwidth) algorithms to mask traffic against traffic classification tools [11], but their technique allows packet fragmentation, and is not applicable to our model.

9 Discussion and Limitations

Improved Dynamic Clustering. When implementing Muffler using dynamic clustering, there remains a choice of how many classes should be derived from the audio corpus. For our testing purposes, we found $k = 32$ to be sufficient for k-means clustering. It may be useful to adjust k given any available background knowledge of the audio streams being combined into a superdistribution. Additionally, other clustering approaches that automate the process of discovering the *number* of clusters (for example, X-means clustering [17]) may serve as a drop-in replacement for k-means clustering.

The Inviability of Pairs. As previously argued, a traffic morphing system that morphs one speaker class to resemble another specific class is not well-suited for masking the identities of a large set of speakers (since it is unlikely that any one speaker in the set will have a distribution to which all other speakers can be padded). However, it could be argued that such approaches are sufficient, when applied in a pairwise fashion. Even if such a method were able to make pairs of speakers indistinguishable, an obfuscation scheme that results in the adversary knowing that a stream comes from one of two speaker classes still leaks considerable information. Additionally, we know that the packet size distributions of the speakers in the pair can very easily be such that one speaker cannot be padded to resemble the other, nor vice versa. This paper argues for a more versatile technique that morphs potentially many input distributions to a single, synthetic target distribution.

Muffler beyond VoIP. This paper shows the effectiveness of Muffler in the context of protecting against VoIP re-identification attacks. The general traffic analysis attack framework applies to other situations in which variations in packet sizes may reveal attributes of the plaintext. For example, similar traffic analysis attacks are applicable to streaming video (which also uses VBR codecs), remote database access, and anonymous web browsing. Although we do not evaluate it in this paper, Muffler can be straightforwardly applied to protect against re-identification attacks on encrypted streaming video. For applications where packets are sent at irregular time intervals—in particular, web browsing—Muffler would also need to consider the *timing* of packets.

10 Conclusion

This paper proposes an efficient blackbox defense called Muffler that protects against encrypted VoIP re-identification attacks. Our approach is based on the fabrication of a *superdistribution* to which all of the streams in a population can be morphed. Experimental results using a large corpus of audio show that even against an adversary who applies a battery of machine learning techniques, Muffler reduces the adversary's accuracy by 94%, while maintaining half of the bandwidth savings provided by using a variable-bitrate codec.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback. This work is partially supported by the National Science Foundation (NSF) through grants CNS-1064986, CNS-1149832, CNS-1445967, and CNS-1223825.

Bibliography

[1] The Speech Accent Archives. URL http://accent.gmu.edu/.

[2] Linguistic Data Consortium (LDC). URL https://www.ldc.upenn.edu/.

[3] Tom Berson. Skype Security Evaluation, October 2005. Available at http://www.anagram.com/berson/abskyeval.html.

[4] A. Bhattacharyya. On a Measure of Divergence between Two Statistical Populations Defined by their Probability Distribution. *Bulletin of the Calcutta Mathematical Society*, pages 99–110, 1943.

[5] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. Touching from a Distance: Website Fingerprinting Attacks and Defenses. In *ACM Conference on Computer and Communications Security (CCS)*, 2012.

[6] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium (USENIX)*, 2004.

[7] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *IEEE Symposium on Security and Privacy (Oakland)*, 2012.

[8] P. Folga, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee. Polymorphic Blending Attacks. In *USENIX Security Symposium*, 2006.

[9] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. Website Fingerprinting: Attacking Popular Privacy Enhancing Technologies with the Multinomial Naïve-bayes Classifier. In *ACM Workshop on Cloud Computing Security (CCSW)*, 2009.

[10] Andrew Hintz. Fingerprinting Websites Using Traffic Analysis. In *Privacy Enhancing Technologies Symposium (PETS)*, 2003.

[11] A. Iacovazzi and A. Baiocchi. Padding and Fragmentation for Masking Packet Length Statistics. In *Traffic Monitoring and Analysis International Workshop*, 2012.

[12] George Kadianakis. Packet Size Pluggable Transport and Trafc Morphing. Technical Report 2012-03-004, Tor Project, Inc., 2012.

[13] LA Khan, MS Baig, and A.M. Youssef. Speaker Recognition from Encrypted VoIP Communications. *Digital Investigation*, 7(1-2):65–73, October 2010.

[14] Marc Liberatore and Brian Neil Levine. Inferring the Source of Encrypted HTTP Connections. In *ACM Conference on Computer and Communications Security (CCS)*, 2006.

[15] Librivox. Librivox: Free Public Domain Audiobooks. URL http://librivox.org/.

[16] Jean Mercler. 50 Million Concurrent Users Online. Available at http://skypenumerology.blogspot.se/2013/01/50-million-concurrent-users-online.html.

[17] D. Pelleg and A. W. Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *International Conference on Machine Learning*, 2000.

[18] Qixiang Sun, Daniel R. Simon, Yi-Min Wang, Wilf Russell, Venkata N. Padmanabhan, and Lili Qiu. Statistical Identification of Encrypted Web Browsing Traffic. In *IEEE Symposium on Security and Privacy (Oakland)*, 2002.

[19] TeleGeography. International Carriers' Traffic Grows Despite Skype Popularity, 2006. Available at `http://www.telegeography.com/products/commsupdate/articles/2006/12/01/telegeography-update-international-carriers-traffic-grows-despite-skype-popularity/`.

[20] TeleGeography. Telegeography Report Executive Summary, 2013. Available at `http://www.telegeography.com/page_attachments/products/website/research-services/telegeography-report-database/0003/6770/TG_executive_summary.pdf`.

[21] K. Vos, S. Jensen, and K. Soerensen. SILK Speech Codec. Internet-Draft draft-vos-silk-02, Internet Engineering Task Force, September 2010.

[22] Tao Wang and Ian Goldberg. Improved Website Fingerprinting on Tor. In *Workshop on Privacy in the Electronic Society (WPES)*, 2013.

[23] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective Attacks and Provable Defenses for Website Fingerprinting. In *USENIX Security Symposium (USENIX)*, 2014.

[24] Andrew M. White, Kevin Snow, Austin Matthews, and Fabian Monrose. Phonotactic Reconstruction of Encrypted VoIP Conversations: Hookt on fon-iks. In *IEEE Symposium on Security and Privacy (Oakland)*, 2011.

[25] I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Burlington, MA, 3rd edition, 2011.

[26] Charles V. Wright, Lucas Ballard, Fabian Monrose, and Gerald M. Masson. Language Identification of Encrypted VoIP Traffic: Alejandra y Roberto or Alice and Bob? In *USENIX Security Symposium (USENIX)*, 2007.

[27] Charles V. Wright, Lucas Ballard, Scott E. Coull, Fabian Monrose, and Gerald M. Masson. Spot Me if You Can: Uncovering Spoken Phrases in Encrypted VoIP Conversations. In *IEEE Symposium on Security and Privacy (Oakland)*, 2008.

[28] Charles V. Wright, Scott E. Coull, and Fabian Monrose. Traffic Morphing: An Efficient Defense Against Statistical Traffic Analysis. In *Network and Distributed System Security Symposium (NDSS)*, 2009.

A Security Analysis

We do not attempt to strengthen the security of VBR encoding to traditional IND-CPA but argue that under certain assumptions, our scheme is able to provide information theoretic indistinguishability against the best known speaker re-identifying attacker.

Definition 1. *A scheme is IND-CGA (Indistinguishability against Chosen Generator Attack) secure if, for all pairs of probabilistic polynomial-time adversaries A_1, A_2, their advantage in the following game is negligible.*

Algorithm 3 Security Experiment

$b \xleftarrow{\$} \{0, 1\}$
$sd \xleftarrow{\$} calcSuperdistribution(trainingData)$
$(g_0, g_1, state) \xleftarrow{\$} A_1(sd, trainingData)$
$stream \xleftarrow{\$} g_b()$
$c \xleftarrow{\$} morphStream(stream, sd, trainingData, state, c)$
$b' \xleftarrow{\$} A_2(sd, trainingData, state, c)$
Return $(b == b')$

The $\xleftarrow{\$}$ notation implies that the function on the right is randomized. In this game, the adversary (the pair of algorithms A_1, A_2) has access to the training data and the superdistribution sd. For simplicity, we consider $numPossibleSizes$ and $gramSize$ fixed and public. The adversary selects two stream generators g_0, g_1, where the generators produce packet streams under some restrictions detailed below. The game selects one at random, generates an actual packet stream from it, morphs it to c using our morphing (Algorithm 2) and returns it to the adversary. The adversary's goal is to determine which generator was selected.

We first define generators.

Definition 2. *Generators model speakers whose audio is processed into packets as a VBR codec encryption layer would. A stream of packets output by a specific generator shares n-gram characteristics with all other streams output by that generator. A generator's output is always randomized in the same way that the audio streams by the same speaker having 2 different conversations will be encoded differently.*

Since we perform a black-box modification of the packet stream by padding it, allowing the adversary to define, and therefore know, the input packet stream will allow it to win the game trivially. By allowing the adversary to define a generator, the adversary is still able to select the stream characteristics which will give it the best probability of winning the game.

While not a rigorous definition, this allows a generator to be implemented as a human speaker who is generating packets by using an encrypted VoIP service, or even a text-to-speech program with a large set of words, where generating output corresponds

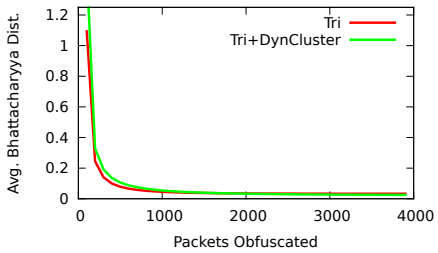

Fig. 10: The Bhattacharyya Distance between morphed distributions and the expected perfect output of the superdistribution

to selecting a random string of words, running them through the text-to-speech program and then running the produced audio through an encrypted VoIP service.

We also make the following assumptions and restrictions, with justification, to complete our security argument.

Assumption A1 *The adversary is only allowed to choose generators whose output characteristics are covered by the training data.*

A generator with output that does not sufficiently match any of the training data corresponds to a speaker whose speech patterns are not represented in the training data. Unfortunately, our system is not designed to protect such users.

Assumption A2 *Our probabilistic morphing technique maps a valid packet stream (one which follows assumption A1) to one which is is negligibly close to the superdistribution.*

Additionally, the output packet stream distribution does not vary over time.

Our morphing algorithm is designed such that the output stream converges to the superdistribution quickly and stays there. To evaluate whether this holds in practice, we compared morphed distributions to the expected output of the superdistribution. We used the Bhattacharyya distance measure [4] which is used to measure the similarity between two discrete (or continuous) probability distributions. This measure has been used in feature extraction and speaker recognition among other areas of research.

To construct the expected trigram distribution of the superdsitribution, we generated packet streams using the superdistribution as a transition matrix. Recall that the superdistribution, on trigrams, is defined as 64 probability distributions, one for each bigram prefix. As such, we generated streams with each of the possible bigrams as an initial state, repeated this process 1000 times, and calculated the expected distribution over all the runs.

Figure 10 shows, for both our labeled and unlabeled techniques, the mean Bhattacharyya distance, over all morphed streams in our corpus against the superdistribution described above, as the number of packets in the stream increases. As the figure shows, the distance quickly converges to 0 as the number of packets obfuscated increases, indicating that the distributions are very similar.

Assumption A3 *In a realistic stream of packets, any long subsequence of packets carries very little, if any, additional information.*

The efficiency of our unigram obfuscator against a tri-gram adversary in Section 7.3 lends support to the assumption that any n-gram characteristics for large n are removed or reduced after morphing.

Under these assumptions, it is straightforward to show that our scheme is IND-CGA (Indistinguishability against Chosen Generator Attack) secure.

Proof. From assumptions A1 and A2, the stream returned to A_2 will have n-gram characteristics of the superdistribution.

We remark that classification with n-grams is the basis for all re-identification attacks with which we are familiar [24, 26–28], and is regularly used in informational retrieval and natural language processing for similar identification tasks. That is, we believe our adversary model reflects best-known attack techniques.

From assumption A3, the returned packet stream is effectively indistinguishable against such an adversary.

The other thing the adversary can do is to attempt to first reverse the morphing before deciding which stream was used. Consider its attempt to revert the ith packet. From assumption A2 and the way the morphing probabilities are calculated, we note that his probabilistic inference on the source packet, based on the what the packet is and all preceeding packets, is always the same (no matter what the source packet actually was) since the n-gram distribution of packets prior to i is the superdistribution. Therefore, the best the adversary can do is guess.

What remains to be shown is how the security argument is affected by relaxing assumption A2. Since the algorithm works as a black box with actual packet streams, it isn't always able to output the packet that would keep the actual output n-gram distribution close to the superdistribution.

We postulate that for short packet streams, where our algorithms works the poorest, the adversary does poorly due to lack of information. On long packet streams the output distribution is very close to the superdistribution, as shown by the bhattacharyya distance tests in Figure 10. As the source streams embed more difficult patterns of n-grams which prevent us from outputting the superdistribution, the adversary's advantage, and the extent of his ability to reverse the morphing, increases.