Reading Between the Lines: Lessons from the SDMI Challenge

Scott A. Craver, John P. McGregor, Min Wu, Bede Liu,
Department of Electrical Engineering, Princeton University

Adam Stubblefield, Ben Swartzlander, Dan S. Wallach,
Department of Computer Science, Rice University

Drew Dean,

Edward W. Felten
Department of Computer Science, Princeton University

Abstract

The Secure Digital Music Initiative is a consortium of parties interested in preventing piracy of digital music, and to this end they are developing architectures for content protection on untrusted platforms. SDMI recently held a challenge to test the strength of four watermarking technologies, and two other security technologies. No documentation explained the implementations of the technologies, and neither watermark embedding nor detecting software was directly accessible to challenge participants. We nevertheless accepted the challenge, and explored the inner workings of the technologies. We report on our results here.

1 Introduction

The Secure Digital Music Initiative (SDMI), a consortium of music-industry companies, is working to develop and standardize technologies that "protect the playing, storing, and distributing of digital music." [4] SDMI has released little information to the public about its technologies.

In September 2000, SDMI announced a "public challenge" in which it invited members of the public to try to break certain data-encoding technologies that SDMI had developed [6]. The challenge offered a valuable window into SDMI, not only into its technologies but also into its plans and goals. We decided to use the challenge to learn as much as we could about SDMI. This paper is the result of our study.[1] Section 2 presents an overview of the SDMI challenge. Section 3 analyzes the watermark challenges. Section 4 analyzes the non-watermark challenges. Finally, we present our conclusions in section 5.

2 The SDMI Challenge

The SDMI challenge extended over roughly a three-week period, from September 15, 2000 until October 8, 2000. The challenge actually consisted of six sub-challenges, named with the letters A through F, each involving a different technology developed by SDMI's members. We believe these challenges correspond to submissions to the SDMI's Call for Proposals for Phase II Screening Technology [5]. According to this proposal, the watermark's purpose is to enforce a usage policy for an audio clip which is compressed or has previously been compressed. That is, an audio clip possessing a watermark may be admitted into an SDMI device, but only if it has not been degraded by compression.

For each challenge, SDMI provided some information about how a technology worked, and then challenged the

[1] The SDMI challenge offered a small cash payment to be shared among everyone who broke one of the technologies according to criteria set by SDMI, and who was willing to sign a confidentiality agreement giving up all rights to discuss their findings. We chose to forgo the payment and retain our right to publish this paper.

public to create an object with a certain property. The exact information provided varied among the challenges, although we note that in all six cases SDMI provided less information than a would-be copyright infringer would have access to in practice.

In addition, the music clips provided by SDMI in connection with the challenge came with a "click-through agreement" that allowed the files to be used only during the three-week challenge period. The limited time allowed for the challenge, and the apparent prohibition on certain follow-up research beyond the three-week period, prevented us from doing more extensive testing, leaving some of our results in an incomplete state. Of course, a would-be copyright infringer presumably would not be deterred by such legal agreements.

2.1 Watermark Challenges

Four of the challenges (A, B, C, and F), involved watermarking technologies, in which subtle modifications are made to an audio file to encode information without perceptible change in how the file sounds. Watermarks can be either *robust* or *fragile:* robust watermarks are designed to survive common transformations like digital-to-audio conversion, compression and decompression, and the addition of small amounts of noise to the file; whereas fragile watermarks do not survive such transformations, and are used to indicate modification of the file.

For each of the four watermark challenges, SDMI provided three files:

- *File 1*: an unwatermarked song;
- *File 2*: File 1, with a watermark added; and
- *File 3*: another watermarked song.

The challenge was to produce a file that sounded just like File 3 but did not have a watermark — in other words, to render the watermark undetectable.

SDMI provided an on-line "oracle" for each challenge. Entrants could send a file to the oracle, and the oracle would inform them if their submission satisfied the challenge, that is, if it contained no detectable watermark while still sounding like File 3. Entrants were given no information about how watermark information was stored in the file or how the oracle detected watermarks, beyond the information that could be deduced from inspection of the three provided files and the oracle's output.

2.2 Challenges D and E

Challenge D concerned a technology designed to prevent a song from being separated from the album in which it was issued. Normally, every Compact Disc contains a table of contents, indicating the offsets and lengths of each audio track, followed by the audio data itself. Challenge D adds an "authenticator" track (approximately 50ms of very quiet audio), derived somehow from the table of contents. The authenticator is supposed to be difficult to compute for an arbitrary CD. Challenge D is discussed in more detail in Section 4.1.

Challenge E involved a technology similar to D, but one which would be immune to the obvious attack on technology D, in which one compiled an unauthorized CD with the same table of contents as an authorized one for which the authenticator track is known. Unfortunately, this challenge was constructed in a way that made it impossible to even start analyzing the technology. SDMI provided an oracle for this challenge, but unfortunately provided no music samples of any kind, so there was no way to determine what the oracle might be testing for.

Given these facts, we decided not to analyze Challenge E. It is discussed briefly in Section 4.2.

3 The Watermarking Schemes

In this section, we describe our attack(s) on each of the four watermark challenges (A,B,C,F). Our success was confirmed by emails received from SDMI's oracles.

Figure 1 provides an overview of the challenge goal. As mentioned earlier, there are three audio files per watermark challenge: an original and watermarked version of one clip, and then a watermarked version of a second clip, from which the mark is to be removed. All clips were 2 minutes long, sampled at 44.1kHz with 16-bit precision.

The reader should note one serious question regarding this challenge arrangement. The challenge is to render a robust mark undetectable, while these technologies appear to be Phase II watermark screening tech-

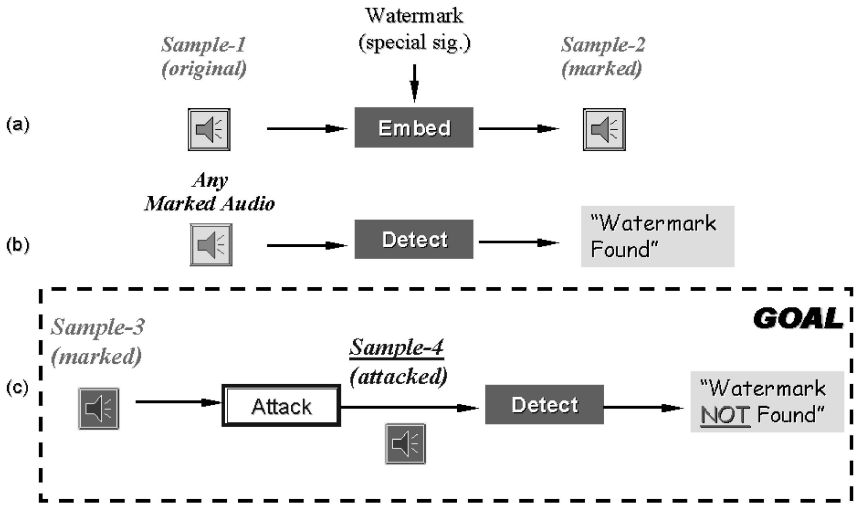

Figure 1: The SDMI watermark attack problem. For each of the four watermark challenges, Sample-1, sample-2, and sample-3 are provided by SDMI. Sample-4 is generated by participants in the challenge and submitted to SDMI oracle for testing.

nologies [5]. As we mentioned above, a Phase II screen is intended to reject audio clips if they have been compressed, and presumably compression degrades a fragile component of the watermark. An attacker need not remove the robust watermark to foil the Phase II screen, but alternatively could repair the modified fragile component in compressed audio. This attack was not available under the challenge setup.

Our analysis of the watermarking schemes uses standard signal processing methods. The text below presumes the reader has a basic understanding of signal processing. Readers without such a background might want to consult a source such as the textbook by Steiglitz [7].

3.1 Attack and Analysis of Technology A

A reasonable first step in analyzing watermarked content with original, unmarked samples is differencing the original and marked versions in some way. Initially, we used sample-by-sample differences in order to determine roughly what kinds of watermarking methods were taking place. Unfortunately, technology A involved a slowly varying phase distortion which masked any other cues in a sample-by-sample difference. We ultimately decided this distortion was a pre-processing separate from the watermark, in part because undoing the distortion alone did not foil the oracle.

The phase distortion nevertheless led us to attempt an attack in which both the phase and magnitude change between sample 1 and sample 2 is applied to sample 3. The attack was based on this code, where FFT computes the Fast Fourier Transform (FFT), and IFFT computes the inverse FFT.

```
while(framesLeftInSong()){
   Y = FFT(nextFrame(markedMusicFile));
   X = FFT(nextFrame(unmarkedMusicFile));
   H = elementwiseDivide(Y,X);
   Z = FFT(nextFrame(otherMarkedFile));
   R = elementwiseDivide(Z,H);
   outputFrame(IFFT(R));
}
```

This attack was confirmed by SDMI's oracle as successful, and illustrates the general attack approach of imposing the difference in an original-watermark pair upon another media clip. Here, the "difference" is taken in the frequency domain rather than the time domain; we suspected a watermark based on frequency-domain modification because of the apparent presence of phase distortion in the watermarking process. Note that this attack did not require much information about the watermarking scheme itself, and conversely did not provide much extra insight into its workings.

A next step, then, is to compute the frequency reponse $H(\omega) = W(\omega)/O(\omega)$ of the watermarking process for segments of audio, where W is the (frequency-domain)

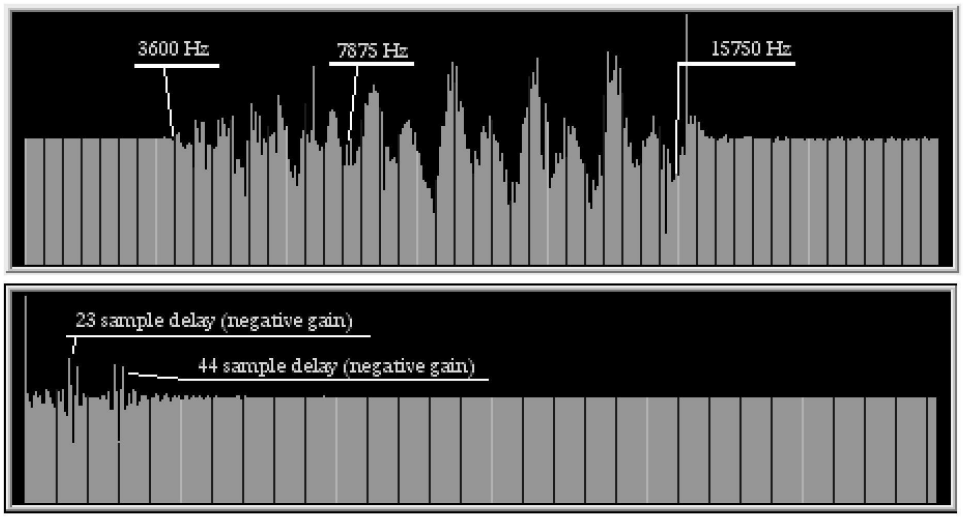

Figure 2: A short-term complex echo. Above, the frequency response between the watermarked and original music, taken over 1/50 second, showing a sinusoidal ripple between 8 and 16 KHz. Below, the corresponding impulse response. The sinusoidal pattern in the frequency domain corresponds to a pair of echoes in the time domain.

watermarked signal and O is the (frequency-domain) original signal, and to observe both $|H(\omega)|$ and the corresponding impulse response $h(t)$. If the watermark is based on some kind of linear filter, whose properties change slowly enough relative to the size of a frame of samples, then this approach is ideal.

Figure 2 illustrates one frequency response and impulse response about 0.3 seconds into the music. These responses are based on FFTs of 882 samples, or one fiftieth of a second of music. As can be clearly seen, a pair of sinusoidal ripples are present within a certain frequency band, approximately 8-16Khz. An echo in the time domain manifests itself as a ripple in the frequency domain, so a sum of sinusoids in the frequency domain suggests the presense of multiple echoes. The corresponding impulse response $h(t)$ confirms this. This pattern of ripples changes quite rapidly from frame to frame.

Thus, we had reason to suspect a complex echo hiding system, involving multiple time-varying echoes. It was at this point that we considered a patent search, knowing enough about the data hiding method that we could look for specific search terms, and we were pleased to discover that this particular scheme appears to be listed as an alternative embodiment in US patent number 5,940,135, awarded to Aris corporation, now part of Verance [3]. This provided us with little more detail than we had already discovered, but confirmed that we were on the right track, as well as providing the probable identity of the company which developed the scheme. It also spurred no small amount of discussion of the validity of Kerckhoffs's criterion, the driving principle in security that one must not rely upon the obscurity of an algorithm. This is, surely, doubly true when the algorithm is patented.

The most useful technical detail provided by the patent was that the "delay hopping" pattern was likely discrete rather than continuous, allowing us to search for appropriate frame sizes during which the echo parameters were constant. Data collection from the first second of audio showed a frame size of approximately 882 samples, or 1/50 second. We also observed that the mark did not begin until 10 frames after the start of the music, and that activity also existed in a band of lower frequency, approximately 4-8 Khz. This could be the same echo obscured by other operations, or could be a second band used for another component in the watermarking scheme. A very clear ripple in this band, indicating a single echo with a delay of about 34 samples, appears shortly before the main echo-hopping pattern begins.

The next step in our analysis was the determination of the delay hopping pattern used in the watermarking method, as this appeared to be the "secret key" of the data embedding scheme. It is reasonable to suspect that the pattern repeats itself in short order, since a watermark detector should be able to find a mark in a subclip of music, without any assistance initially aligning the mark with the detector's hopping pattern. Again, an analysis of the first second revealed a pattern of echo pairs that appeared to repeat every 16 frames, as outlined in figure 3. The delays appear to fall within six general categories, each

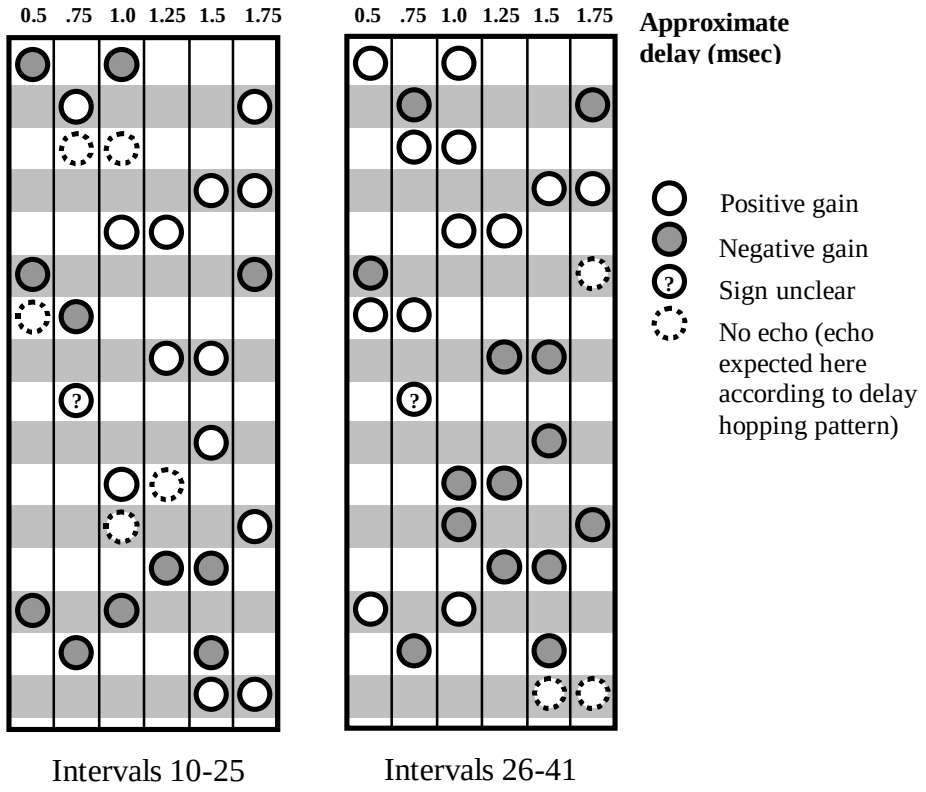

Figure 3: The hypothesized delay hopping pattern of technology A. Here two stretches of 16 frames are illustrated side-by-side, with observed echoes in each frame categorized by six distinct delays: 2, 3, 4, 5, 6 or 7 times 0.00025 sec. Aside from several missing echoes, a pattern appears to repeat every 16 frames. Note also that in each frame the echo gain is the same for both echoes.

delay approximately a multiple of 1/4 millisecond. The exact values of the delays vary slightly, but this could be the result of the phase distortion present in the music.

The reader will also note that in apparently two frames there is only one echo. We found two possible explanations for this. First, if we made two independent random choices from six possible echo delays for each frame, then we would expect the two random choices to coincide (i.e., to be equal to each other) about one-sixth of the time; the observed ratio of two coincidences in sixteen frames is consistent with this hypothesis. Second, our detector appears to have missed some echoes, which show up in one of the two sides of the figure but not the other. (These are depicted as dotted circles on the side where they appear to have been missed.) If the detector happened to miss the same echo in *both* sides of the figure, this would lead to a frame that appeared to have only one echo. Again, the frequency of missing echoes is roughly consistent with this hypothesis. Of course, it may be that some other explanation is correct.

Next, there is the issue of the actual encoded bits. Further work shows the sign of the echo gain does not repeat with the delay-hopping pattern, and so is likely at least part of an embedded message. Extracting such data without the help of an original can be problematic, although the patent, of course, outlines numerous detector structures which can be used to this end. We developed several tools for cepstral analysis to assist us in the process. See [2] for in introduction to cepstral analysis; Anderson and Petitcolas [1] illustrate its use in attacks on echo hiding watermark systems.

With a rapidly changing delay, normal cepstral analysis does not seem a good choice. However, if we know that the same echo is likely to occur at multiples of 16/50 of a second, we can improve detector capability by combining the information of multiple liftered[2] log spectra.

[2] In accordance with the flopped vocabulary used with cepstral analysis, "liftering" refers to the process of filtering data in the frequency domain rather than the time domain. Similarly, "quefrencies" are frequencies of ripples which occur in the frequency domain rather than the time domain.

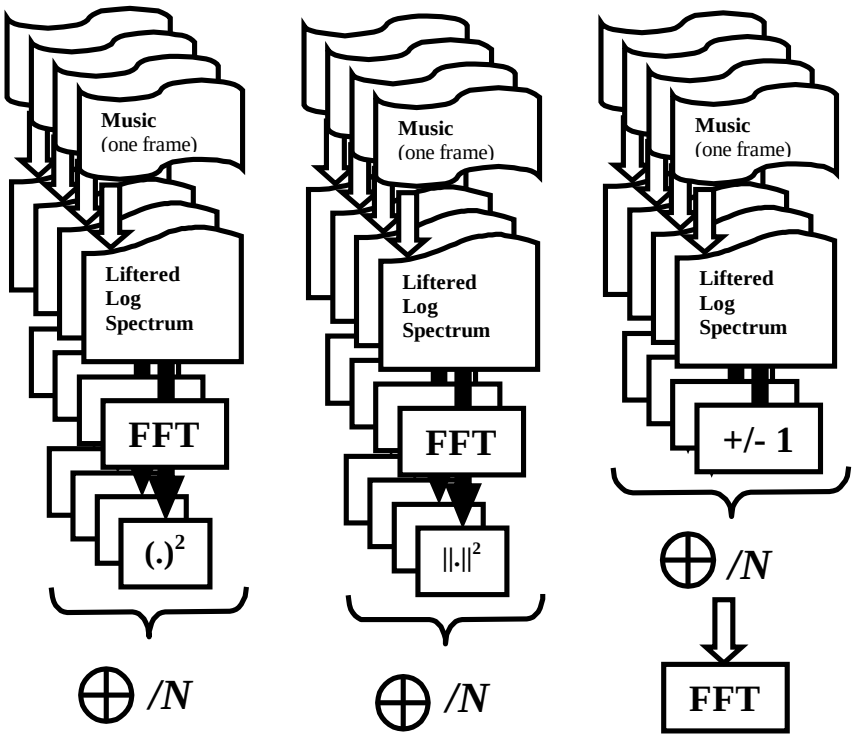

Figure 4: Three cepstral detector structures. In each case we have a collection of distinct frames, each believed to possess echoes of the same delay. The first two compute cepstral data for each frame, and sum their squares (or squared magnitudes) to constructively combine the echo signal in all frames. The third structure illustrates a method for testing a hypothesized pattern of positive and negative gains, possibly useful for brute-forcing or testing for the presence of a known "ciphertext."

Three detector structures are shown in figure 4. In all three, a collection of frames are selected for which the echo delays are believed to be the same. For each, the liftered log of an FFT or Power Spectral Distribution (PSD) of the frame is taken. In the first two structures, we compute a cepstrum for each frame, then either average their squared magnitudes, or simply their squares, in hopes that a spike of the appropriate quefrency will be clear in the combination. The motivation for merely squaring the spectral coefficients comes from the observation that a spike due to an echo will either possess a phase of ϕ or $\phi + \pi$ for some value ϕ. Squaring without taking magnitudes can cause the echo phases to reinforce, whilst still permitting other elements to combine destructively.

In the final structure, one cepstrum is taken using a guess of the gain sign for each suspect frame. With the correct guess, the ripple should be strongest, resulting in the largest spike from the cepstral detector. Figure 5 shows the output of this detector on several sets of suspect frames. While this requires an exponential amount of work for a given number of frames, it has a different intended purpose: this is a brute-forcing tool, a utility for determining the most probable among a set of suspected short strings of gain signs as an aid to extracting possible ciphertext values.

Finally, there is the issue of what this embedded watermark means. Again, we are uncertain about a possible signalling band below 8KHz. This could be a robust mark, signalling presence of a fragile mark of echoes between 8 and 16 KHz. The 8-16KHz band does seem like an unusual place to hide robust data, since compressors often reduce the information in this band greatly. If the signal does indeed extend further down, the 8-16KHz band could very easily be hidden information whose degradation is used to determine if music has already been compressed.

Of course, knowledge of *either* the robust or fragile component of the mark is enough for an attacker to defeat the scheme, because one can either remove the robust mark, or repair or reinstate the fragile mark after compression

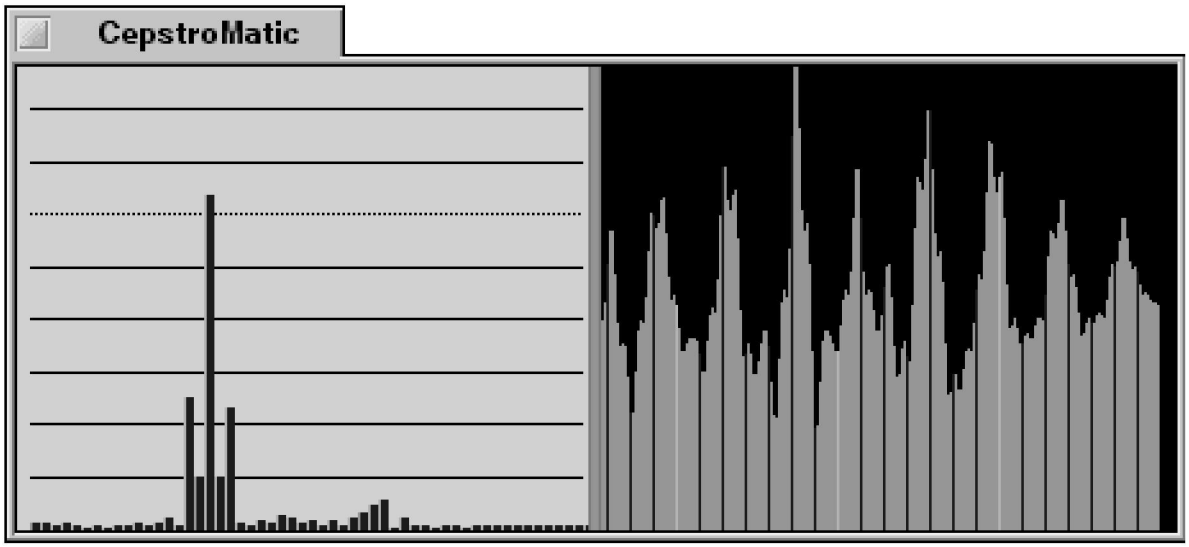

Figure 5: Detection of an echo. A screenshot of our CepstroMatic utility shows a combination of 4 separate frames of music, each a fiftieth of a second long, in which the same echo delay was believed to exist. Their combination shows a very clear ripple on the right, corresponding to a clear cepstral spike on the left. This is a single echo at a delay of 33 samples, the delay suggested for these intervalus by the hypothesized delay-hopping pattern.

has damaged it. As mentioned earlier, this possible attack of repairing the fragile component appears to have been ruled out by the nature of the SDMI challenge oracles. One must wait and see if real-world attackers will attempt such an approach, or resort to brute force methods or oracle attacks to remove the robust component.

3.2 Attack on Challenge B

For Challenge B, we analyzed the matching unwatermarked and watermarked samples using a short-time FFT. Shown in Fig. 6 are the two FFT magnitudes for 1000 samples at 98.67 sec. Also shown is the difference of the two magnitudes. A spectrum notch around 2800Hz is observed for some segments of the watermarked sample and another notch around 3500Hz is observed for some other segments of the watermarked sample. Similar notches are observed in the other watermarked sample. The attack fills in those notches with random but bounded coefficient values. We also submitted a variation of this attack involving different parameters for notch description. Both attacks were confirmed by the SDMI oracle as successful.

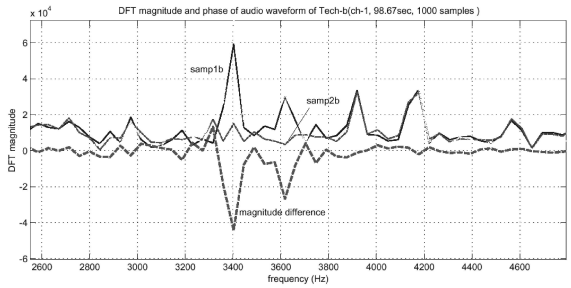

Figure 6: Challenge B: FFT magnitudes of the matching watermarked (samp1b) and unwatermarked (samp2b) files and their difference for 1000 samples at 98.67 sec.

3.3 Attacks on Challenge C

By taking the difference between the matching watermarked and unwatermarked samples for Challenge C, we observed bursts of narrowband signal, as shown in Fig. 7. These narrow band bursts appear to be centered around 1350 Hz, suggesting that the detector is looking for something to be present at 1350 Hz. We applied two different attacks to Challenge C. In the first attack, we shifted the pitch of the audio by about a quartertone to move the bursts away from 1350 Hz. In the second attack, we passed the signal through a bandstop filter centered around 1350Hz. Both submissions were confirmed

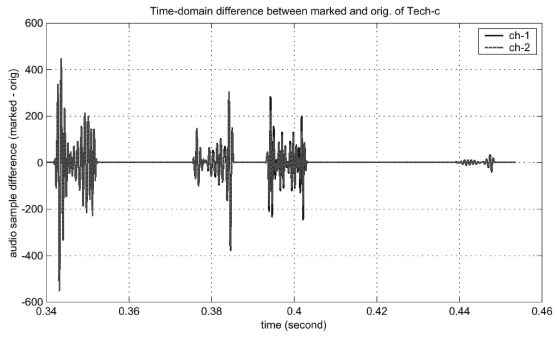

Figure 7: Challenge C: Waveform of the difference between the matching watermarked and unwatermarked files.

by SDMI oracle as successful. In addition, the perceptual quality of both attacks passed the "golden ear" testing conducted by SDMI after the 3-week challenge.

3.4 Attack on Challenge F

For Challenge F, we warped the time axis, by inserting a periodically varying delay. Essentially, we computed $y(t) = x(t + f(t))$, in which x is the original audio signal and f is a slowly-varying sinusoidal function. In our first successful attack, $f(t) = 3(1 - cos(0.602\pi t))^2/19600$, where t is measured in seconds. This has a period of approximately 3.32 seconds, and distorts the music by a maximum of 27 samples, or about 0.6 milliseconds. The attack is implemented by the code shown in Figure 8.

The delay function comes from our study of Technology A, and was in fact initially intended to undo the phase distortion applied by Technology A. Therefore, the perceptual quality of our attacked audio is expected to be better than or comparable to that of the audio watermarked by Technology A. We also submitted variations of this attack involving different warping parameters and different delay functions. They were confirmed by the SDMI oracle as successful.

4 The Non-Watermark Technologies

The SDMI challenge contained two "non-watermark" technologies. Together, they appear to be intended to prevent the creation of "mix" CDs, where a consumer might compile audio files from various locations to a

```
int main(int argc, char* argv[]){
   int k, numSamples;
   int left,right,oldleft,oldright;
   int cumulative = 0;
   double delay;

   numSamples = getNumSamples();
   for(k=0; k<numSamples; ++k){
      readsample(&left, &right);
      delay = 1.0-cos(0.602*PI*k/44100);
      delay = 6.75*delay*delay;
      if((int)delay < cumulative){
         cumulative--;
         writesample(oldleft, oldright);
         writesample(left, right); break;
      }else if((int)delay > cumulative){
         cumulative++;
      }else{
         writesample(left, right);
      }
      oldleft = left; oldright = right;
   }
   return 0;
}
```

Figure 8: Code for the time-axis warping attack on Challenge F.

writable CD. This would be enforced by universally embedding SMDI logic into consumer audio CD players.

4.1 Technology D

According to SDMI, Technology D was designed to require "the presence of a CD in order to 'rip' or extract a song for SDMI purposes." The technology aimed to accomplish this by adding a 53.3 ms audio track (four blocks of CD audio), which we will refer to as the **authenticator**, to each CD. The authenticator, combined with the CD's table of contents (TOC), would allow an SDMI device to recognize SDMI compliant CDs. For the challenge, SDMI provided 100 different "correct" TOC-authenticator pairs as well as 20 "rogue tracks". A rogue track is a track length that does not match any of the track lengths in the 100 provided TOCs. The goal of the challenge was to submit to the SDMI oracle a correct authenticator for a TOC that contained at least one of the rogue tracks.

The oracle for Technology D allowed several different

query types. In the first type, an SDMI provided TOC-authenticator combination is submitted so that a user can "understand and verify the Oracle." According to SDMI, the result of this query should either be "admit" for a correct pair or "reject" for an incorrect pair. When we attempted this test with an SDMI-provided pair, the oracle responded that the submission was "invalid." After verifying that we had indeed submitted a correct pair, we attempted several other submissions using different TOC-authenticator pairs as well as different browsers and operating systems[3]. We also submitted some pairs that the oracle should have rejected; these submissions were also declared "invalid." Though we alerted SDMI to this problem during the challenge, the oracle was never repaired. For this reason, our analysis of Technology D is incomplete and we lack definitive proof that it is correct. That having been said, we think that what we learned about this technology, even without the benefit of a correctly functioning oracle, is interesting.

4.1.1 Analyzing the Signal

Upon examination of the authenticator audio files, we discovered several patterns. First, the left and right channels contain the same information. The two channels differ by a "noise vector" **u**, which is a vector of small integer values that range from -8 and 8. Since the magnitude of the noise is so small, the noise vector does not significantly affect the frequency characteristics of the signal. The noise values appear to be random, but the noise vector is the same for each of the 100 provided authenticator files. In other other words, in any authenticator file, the difference between the left and right channels of the ith sample is a constant fixed value $\mathbf{u}[i]$. This implies that the noise vector **u** does not encode any TOC-specific information.

Second, the signal repeats with a period of 1024 samples. Because the full signal is 2352 samples long, the block repeats approximately 1.3 times. Similarly to the left and right channels of the signal, the first two iterations of the repeating signal differ by a constant noise vector **v**. The difference between the ith sample of the first iteration and the ith sample of the second iteration differ by a small (and apparently random) integer value $\mathbf{v}[i]$ ranging from -15 to 15. In addition, **v** is the same for each of the provided authenticator files, so **v** does not encode any TOC-specific information.

[3]Specifically, Netscape Navigator and Mozilla under Linux, Netscape Navigator under Windows NT, and Internet Explorer under Windows 98 and 2000.

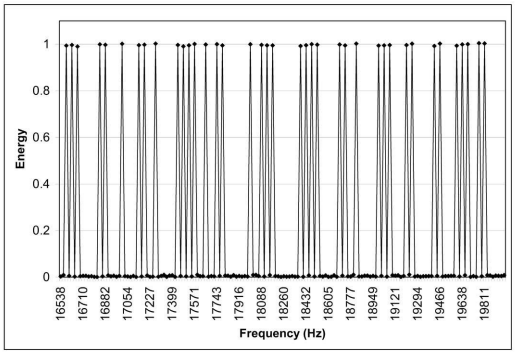

Figure 9: Individual Bits From a Technology D Authenticator

Third, the first 100 samples and last 100 samples of the full signal are faded in and faded out, respectively. The fade-in and fade-out are meaningless, however, because they simply destroy data that is repeated in the middle of the file. We conjecture that this fade-in and fade-out are included so that the audio signal does not sound offensive to a human ear.

4.1.2 Extracting the Data

Frequency analysis on the 1024 sample block shows that almost all of the signal energy is concentrated in the 16–20kHz range. We believe this range was chosen because these frequencies are less audible to the human ear. Closer examination shows that this 16–20kHz range is divided up into 80 discrete bins, each of which appears to carry one bit of information. As shown in Figure 9, these bits can be manually counted by a human using a graph of the magnitude of signal in the frequency domain.

Close inspection and pattern matching on these 80 bits of information reveals that there are only sixteen bits of information repeated five times using different permutations. Using the sixteen letters A-P to symbolize the sixteen bits, these five permutations are described in Figure 10.

Because of the malfunctioning oracle, we were unable to determine the function used to map TOCs to authenticators, but given an actual SDMI device, it would be trivial to brute force all 2^{16} possibilities. Likewise, without the oracle, we could not determine if there was any other signal present in the authenticator (*e.g.,* in the phase of

ABCDEFGHIJKLMNOP
OMILANHGPBDCKJFE
PKINHODFMJBCAGLE
FCKLGMEPNOADJBHI
PMGHLECAKDONIFJB

Figure 10: The encoding of the 16 bits of data in Technology D

the frequency components with nonzero magnitude).

For the moment, let us assume that the hash function used in Technology D has only 16 bits of output. Given the number of distinct CDs available, an attacker should be able to acquire all, or almost all, of the authenticators. We note that at 9 kilobytes each, a collection of 65,536 files would fit nicely on a single CD. Many people have CD collections of 300+ discs, which by the birthday paradox makes it more likely than not that there is a hash collision among their own collection.

Our results indicated that the hash function used in Technology D could be weak or may have less than 16 bits of output. In the 100 authenticator samples provided in the Technology D challenge, there were two pairs of 16-bit hash collisions. We will not step through the derivation here, but for $n < X$, the probability of two or more collisions occurring in n samples of X equally likely possibilities is:

$$1 - \left(\prod_{i=1}^{n-1} \frac{X+1-i}{X} \right) \left(1 + \frac{n^2 - 3n + 2}{2X} \right)$$

If the 16-bit hash function output has 16 bits of entropy, the probability of 2 collisions occurring in $n = 100$ samples of $X = 2^{16}$ possibilities is 0.00254 (by the above equation). If $X \approx 2^{11.5}$, the chances of two collisions occurring is about even. This suggests that either 4 bits of the 16-bit hash output may be outputs of functions of the other 12 bits or the hash function used to generate the 16-bit signature is weak. It is also possible that the challenge designers purposefully selected TOCs that yield collisions. The designers could gauge the progress of the contestants by observing whether anyone submits authenticator A with TOC B to the oracle, where authenticator A is equal to authenticator B. Besides the relatively large number of collisions in the provided authenticators, it appears that there are no strong biases in the authenticator bits such as significantly more or fewer 1's than 0's.

4.2 Technology E

Technology E is designed to fix a specific bug in Technology D: the TOC only mentions the *length* of each song but says nothing about the contents of that song. Accordingly, an attacker wishing to produce a mix CD would only need to find a TOC approximately the same as that of the desired mix CD, then copy the TOC and authenticator from that CD onto the mix CD. If the TOC does not perfectly match the CD, the track skipping functionality will still work but will only get "close" to track boundaries rather than reaching them precisely. Likewise, if a TOC specified a track length longer than the track we wished to put there, we could pad the track with digital silence (or properly SDMI-watermarked silence, copied from another valid track). Regardless, a mix CD played from start to end would work perfectly. Technology E is designed to counter this attack, using the audio data itself as part of the authentication process.

The Technology E challenge presented insufficient information to be properly studied. Rather than being given the original audio tracks (from which we might study the unspecified watermarking scheme), we were instead given the tables of contents for 1000 CDs and a simple scripting language to specify a concatenation of music clips from any of these CDs. The oracle would process one of these scripts and then state whether the resulting CD would be rejected.

While we could have mounted a detailed statistical analysis, submitting hundreds or thousands of queries to the oracle, we believe the challenge was fundamentally flawed. In practice, given a functioning SDMI device and actual SDMI-protected content, we could study the audio tracks in detail and determine the structure of the watermarking scheme.

We later received hints that Technology E may have been susceptible to attack despite the very limited information we were given. If true, this suprising assertion may in itself convey information about how Technology E works. Unfortunately, we did not receive these hints until after the challenge was over, when we no longer had access to the oracle to study the matter further.

5 Conclusion

In this paper, we have presented an analysis of the technology challenges issued by the Secure Digital Music

Initiative. Each technology challenge described a specific goal (*e.g.*, render undetectable a watermark from an audio track) and offered a Web-based oracle that would confirm whether the challenge was successfully defeated. We have defeated all four of their audio watermarking technologies, and have studied and analyzed their "non-watermarking" technologies to the best of our abilities given the lack of information available to us and given a broken oracle in one case.

Some debate remains as to whether our attacks damaged the audio beyond standards measured by "golden ear" human listeners. Given a sufficient body of SDMI-protected content using the watermark schemes presented here, we are confident we could refine our attacks to introduce distortion no worse than the watermarks themselves introduce to the the audio. Likewise, debate remains on whether we have truly defeated technologies D and E. Given a functioning implementation of these technologies, we are confident we can defeat them.

References

[1] ANDERSON, R. J., AND PETITCOLAS, F. A. P. On the limits of steganography. *IEEE Journal of Selected Areas in Communications 16*, 4 (May 1998), 474–481.

[2] BOGERT, R. P., HEALY, M. J., AND TUKEY, J. W. The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-ceptsrum and saphe-cracking. In *Proceedings of the Symposium on Time Series Analysis* (Brown University, June 1962), pp. 209–243.

[3] PETROVIC, R., WINOGRAD, J. M., JEMILI, K., AND METOIS, E. Apparatus and method for encoding and decoding information in analog signals, Aug. 1999. US Patent No. 5,940,135.

[4] SECURE DIGITAL MUSIC INITIATIVE. `http://www.sdmi.org`.

[5] SECURE DIGITAL MUSIC INITIATIVE. *Call for Proposals for Phase II Screening Technology, Version 1.0*, Feb. 2000. `http://www.sdmi.org/download/FRWG00022401-Ph2_CFPv1.0.PDF`.

[6] SECURE DIGITAL MUSIC INITIATIVE. SDMI public challenge, Sept. 2000. `http://www.hacksdmi.org`.

[7] STEIGLITZ, K. *A Digital Signal Processing Primer: with Applications to Digital Audio and Computer Music*. Addison Wesley, 1996.