Analysis of IS-95 CDMA Voice Privacy

Muxiang Zhang[1], Christopher Carroll[1], and Agnes Chan[2]

[1] GTE Laboratories Inc., 40 Sylvan Road LA0MS59, Waltham, MA 02451
{mzhang, ccarroll}@gte.com
[2] College of Computer Science, Northeastern University, Boston, MA 02115
ahchan@ccs.neu.edu

Abstract. The voice privacy of IS-95 CDMA cellular system is analyzed in this paper. By exploiting information redundancy on the downlink traffic channel, it is shown that an eavesdropper can recover the voice privacy mask after eavesdropping the transmission on the downlink traffic channel for about one second. Thus, IS-95 CDMA voice privacy is vulnerable under ciphertext-only attacks.

1 Introduction

IS-95 *Code Division Multiple Access* (CDMA) is an interim industry standard [3] for cellular communication systems. In IS-95 CDMA, a pseudo-random pattern, which is a high bit-rate binary sequence known as *long code sequence*, is added to the low bit-rate voice signal. Adding a high bit-rate noise-like signal to a voice signal makes the voice signal more robust and less susceptible to interference. It enables low-power transmission to take place, resulting in cheaper mobile stations with long-lasting battery life. The long code sequence, which is only known to the designated receiver, is also expected to provide certain level of privacy to the voice signal. To decode the voice signal, the eavesdropper has to recover the long code sequence from the intercepted signal. The long code sequence is generated by the *long code generator* as shown in Figure 3. The long code generator consists of a 42-bit number called *long code mask* and a 42-bit linear feedback shift register (LFSR) specified by the following characteristic polynomial:

$$x^{42}+x^{35}+x^{33}+x^{31}+x^{27}+x^{26}+x^{25}+x^{22}+x^{21}+x^{19}$$
$$+x^{18}+x^{17}+x^{16}+x^{10}+x^{7}+x^{6}+x^{5}+x^{3}+x^{2}+x+1.$$

The inner product of the LFSR state and the long code mask produces the long code sequence.

Voice privacy of IS-95 CDMA is provided by means of the long code mask. The long code mask is not transmitted through any channel, it is constructed by the base station and the mobile station. To recover the long code sequence, the eavesdropper may exhaustively search the 42-bit long code mask, with a time complexity of $O(2^{42})$. This attack is viable but is hard to implement in real time. Alternatively, it can be shown that the long code sequence can also

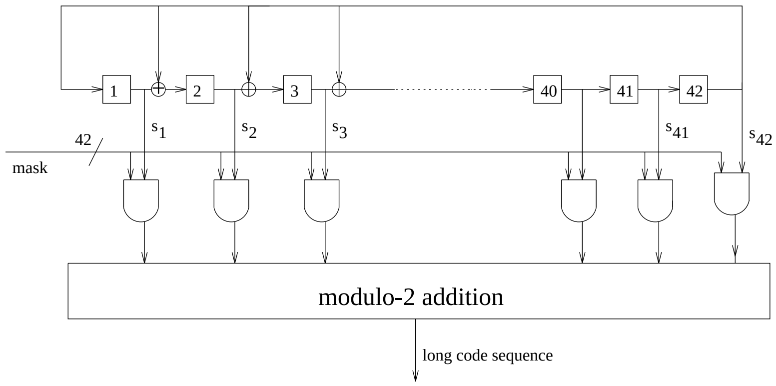

Fig. 1. The long code generator

be recovered if the eavesdropper can obtain 42 bits of plaintext-ciphertext pairs. As there are many mobile stations transmitting simultaneously on the traffic channel and each mobile station only transmits approximately 3 minutes on the average, it is rather difficult to obtain 42 bits of the plaintext message. In this paper, we investigate the security of IS-95 CDMA voice privacy under ciphertext-only attacks. By exploiting information redundancy introduced by channel coding, it is shown that the eavesdropper can recover the voice privacy mask after eavesdropping the transmission on the traffic channel for about one second. This paper is organized as follows. Section 2 gives an equivalent long code generator which can simplify the analysis of CDMA voice privacy. Section 3 describes two kinds of ciphertext-only attacks and Section 4 presents some concluding remarks.

2 An Equivalent Long Code Generator

Let $(m_1, m_2, ...m_{42})$ denote the 42-bit mask and $(s_1(k), s_2(k), ..., s_{42}(k))$ denote the state of the LFSR at time instant k. Then the long code sequence $c(k)$ at time instant k can be represented as

$$c(k) = m_1 s_1(k) + m_2 s_2(k) + \cdots + m_{42} s_{42}(k), \tag{1}$$

where the addition is the modulo-2 addition. Since $s_1(k), s_2(k), \cdots, s_{42}(k)$ are the outputs of the 42 stages of the LFSR, they are the same sequence but only differ in the phase. So, for any i, $1 \leq i \leq 42$, $s_i(k)$ satisfies the following linear recurrence equation,

$$s_i(k) = a_1 s_i(k-1) + a_2 s_i(k-2) + \cdots + a_{42} s_i(k-42), \tag{2}$$

where a_i is the coefficient of x^i in the characteristic polynomial of the LFSR. Substituting (2) into (1), we have

$$\begin{aligned} c(k) &= \sum_{i=1}^{42} m_i s_i(k) \\ &= \sum_{i=1}^{42} m_i (\sum_{j=1}^{42} a_j s_i(k-j)) \\ &= \sum_{i=1}^{42} a_i (\sum_{j=1}^{42} m_j s_j(k-i)). \end{aligned}$$

By equation (1),

$$\sum_{j=1}^{42} m_j s_j(k-i) = c(k-i).$$

Hence, it follows that

$$c(k) = a_1 c(k-1) + a_2 c(k-2) + \cdots + a_{42} c(k-42), \tag{3}$$

which means that the long code sequence is also generated by the same LFSR. Although every mobile station uses a different long code mask, their long code sequences are the same except for the different phases. The long code mask only affects the phase of the long code sequence.

Let $C(k) = (c(k), c(k-1), \cdots, c(k-41))$ and $C(k+1) = (c(k+1), c(k), \cdots, c(k-42))$ be two consecutive states of the LFSR. By equation (3), we have

$$\begin{bmatrix} c(k+1) \\ c(k) \\ \\ \vdots \\ \vdots \\ \\ c(k-42) \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_{40} & a_{41} & a_{42} \\ 1 & 0 & 0 & & 0 & 0 & 0 \\ 0 & 1 & 0 & & 0 & 0 & 0 \\ 0 & 0 & 1 & & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} c(k) \\ c(k-1) \\ \\ \vdots \\ \vdots \\ \\ c(k-41) \end{bmatrix}. \tag{4}$$

Let A^t denote the 42×42 matrix of the above equation and a^t denote the vector $(a_1, a_2, \cdots, a_{42})$, where A^t is the transpose of A. Then for any $n > k$,

$$C(n) = C(k) A^{n-k}.$$

By equation (4), it follows that

$$c(n) = C(k) A^{n-k-1} a. \tag{5}$$

Equation (5) indicates that the state $C(k)$ of the LFSR can be calculated if any 42 bits of the long code sequence are known. Since the LFSR generating the long code sequence is publicly known, the coefficients $a_i, 1 \leq i \leq 42$ are available to the eavesdropper. Thus, the long code sequence can be generated if the eavesdropper can recover 42 bits of the long code sequence.

3 Attacks on CDMA Voice Privacy

Before the analysis of IS-95 CDMA voice privacy, let us give a simple description of CDMA channels. The CDMA channels consists of traffic channels and control channels. The traffic channels carry user information while the control channels carry signalling information. The traffic channels can be further divided into downlink and uplink traffic channels.

3.1 CDMA Traffic Channels

The downlink channel carries information from the base station to the mobile station. On the downlink traffic channel, the vocoder accepts the voice signal and produces a compressed data stream. IS-95 CDMA specifies a variable-rate vocoder operating at full, $\frac{1}{2}$, $\frac{1}{4}$, and $\frac{1}{8}$ rates. The rate is determined according to the power level of the background noise. There are currently two types of vocoders: the one operating in a 9.6-kbps, and the other operating in a 14.4-kbps, referred to as rate set 1 and rate set 2, respectively. Rate set 1 contains four elements: 9.6, 4.8, 2.4, and 1.2 kbps. Rate set 2 also contains four elements: 14.4, 7.2, 3.6, and 1.8 kbps. The mobile station has to support rate set 1, while rate set 2 is optional.

The data stream from the vocoder is structured in 20-ms frames. The full rate of rate set 1 vocoder is 8.6 kbps, which generates 172 bits every 20 ms. The frame quality indicator, which is cyclic redundancy checking (CRC) digits derived from the 172 information bits, is added to the 172 bits along with an 8-bit tail (set to 0). The 9.6 kbps frame is a result of 192 bits (172+12+8) transmitted every 20 ms. The 4.8-kbps frame has the same structure, while 2.4- and 1.2-kbps frames do not have frame quality indicator fields since most of the information sent in these frames is background noise. Rate set 2 frames have similar structures as rate set 1. The 20-ms data frames are encoded, interleaved, scrambled, spread, and modulated before they are sent onto the air interface. Figure 2 shows the different functions which act on rate set 1 data frames.

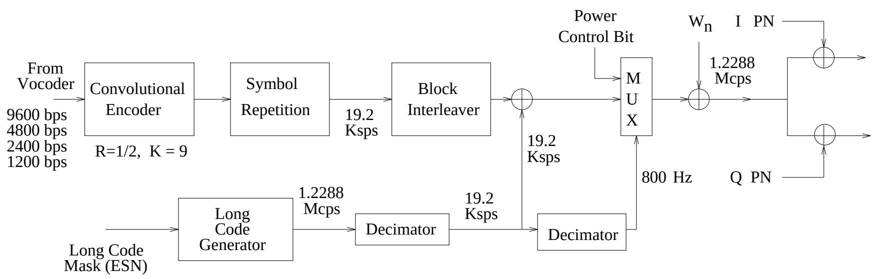

Fig. 2. Rate set 1 downlink traffic channel generation

The convolutional encoder provides error-correction capability to the downlink traffic channels. The convolutional encoder replace each single input bit with two bits (called code symbols). The symbol repeater repeats the code symbols produced by the convolutional encoder as necessary to result in an output stream with the fixed rate 19.2 ksps (kilo-symbols per second). For example, to achieve this, it does not repeat anything if the input rate is 19.2 ksps. It repeats each symbol twice if the input rate is 9.6 ksps, it repeats each symbol 4 times if the input rate is 4.8 ksps. The block Interleaver shuffles the code symbols in each data frame. Data scrambling is provided through the long code generator. A power control subchannel is continuously transmitted on the downlink traffic channel. It is used to control the mobile station's power on the uplink. This subchannel transmits at a rate of one bit every 1.25 ms (i.e., 800 bps). The power control bit which is two symbols long replaces two consecutive symbols on the downlink traffic channel. The 19.2 ksps data stream, which has been punctured with the power control bits of the power control subchannel, is spread by a Walsh code. Following the Walsh code spreading, the data is modulated for transmission.

Like the downlink traffic channel, the uplink traffic channel also supports two rate sets, depending on the type of vocoder used. Figure 3 shows the overall structure of the uplink traffic channel for rate set 1.

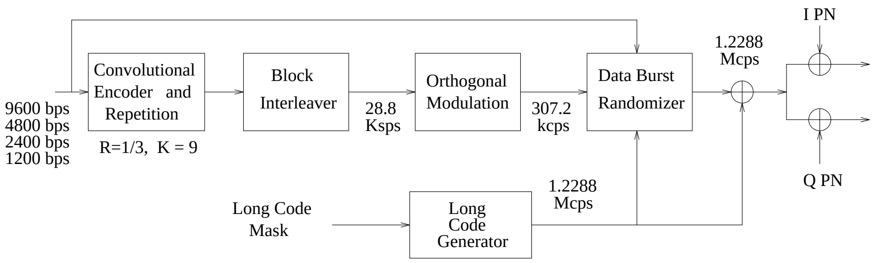

Fig. 3. Rate set 1 uplink traffic channel generation

3.2 Information redundancy on the downlink traffic channel

On the downlink traffic channel, information bits from the vocoder are coded by a convolution encoder as shown in Figure 4. The convolutional encoder has 1-bit input, 2-bit output, and 8-bit memory. Initially the 8 memory bits are filled with all 0s. For every bit of input, the convolutional encoder outputs 2 bits. In coding theory, each bit of output is called a code symbol. Such a convolutional encoder is called a half-rate convolutional encoder [1]. Because of the 8-bit memory, each code symbol is related to 9 information bits. The number 9 is called the constraint length of the convolutional encoder. The half-rate convolutional encoder with constraint length of 9 is also called a (2, 1, 8) convolutional encoder.

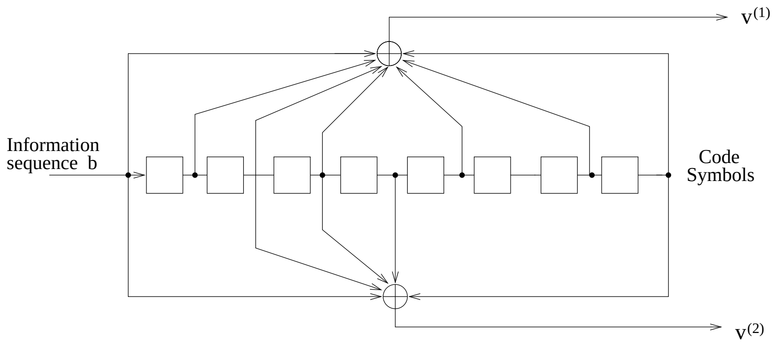

Fig. 4. Half-rate convolutional encoder

Let $b = (b_0, b_1, b_2, \ldots)$ denote the input sequence entering the convolutional encoder. The two output sequences

$$v^{(1)} = (v_0^{(1)}, v_1^{(1)}, v_2^{(1)}, \ldots)$$

and

$$v^{(2)} = (v_0^{(2)}, v_1^{(2)}, v_2^{(2)}, \ldots)$$

can be computed as follows:

$$v_l^{(1)} = b_l + b_{l-1} + b_{l-2} + b_{l-3} + b_{l-5} + b_{l-7} + b_{l-8}, \tag{6}$$

$$v_l^{(2)} = b_l + b_{l-2} + b_{l-3} + b_{l-4} + b_{l-8}. \tag{7}$$

where $b_{l-i} = 0$ for all $l < i$. The two output sequences are multiplexed into a single sequence, called the code word. Let $v = (v_0^{(1)}, v_0^{(2)}, v_1^{(1)}, v_1^{(2)}, v_2^{(1)}, v_2^{(2)}, \ldots)$ denote the code word. From (6) and (7), the code word satisfies the following equation:

$$vH = 0, \tag{8}$$

where H is a semi-infinite matrix given by

$$H = \begin{bmatrix} 1\,0\,1\,1\,1\,0\,0\,0\,1 & & & \\ 1\,1\,1\,1\,0\,1\,0\,1\,1 & & & \\ & 1\,0\,1\,1\,1\,0\,0\,0\,1 & & \\ & 1\,1\,1\,1\,0\,1\,0\,1\,1 & & \\ & & \vdots\;\vdots & \\ & & 1\,0 & \cdots \\ & & 1\,1 & \cdots \\ & & & \ddots \end{bmatrix}.$$

In equation (8), the code word is treated as a semi-infinite sequence since the input sequence to the convolutional encoder may be semi-infinite. When the input sequence to the convolutional encoder is truncated to k bits and the last 8 bits are all zero, the code symbol sequence has length $n = 2k$ and satisfies the following equation:

$$v[H]_n = 0, \tag{9}$$

where $[H]_n$ is composed of the first n rows and $n/2 + 8$ columns of H.

Recall that the information bits entering the convolutional encoder are structured in 20-ms frames and the last 8 bits of each frame are set to zero. As a consequence, the code symbols in every 20-ms frame satisfy equation (9). Thus, every frame on the downlink traffic channel contains redundant information. The redundant information can be used to solve for the long code sequence.

Except the convolutional encoding, symbol repetition also incurs redundant information. The code symbols from the convolutional encoder are repeated whenever the information rate is lower than 9.6 kbps. Every code symbol at the 4.8 kbps rate is repeated 1 time, every code symbol at the 2.4 kbps rate is repeated 3 times, and every code symbol at the 1.2 kbps rate is repeated 7 times. Every frame contains 384 code symbols after repetition. Let $u = (u_1, u_2, \cdots, u_{384})$ denote the frame after code symbol repetition. If each code symbol appears two times in u, it is obvious that

$$uE_2 = 0, \tag{10}$$

where E_2 is a matrix described by

$$E_2 = \begin{bmatrix} 1\,0\,0 \cdots 0 \\ 1\,0\,0 \cdots 0 \\ 0\,1\,0 \quad\;\; 0 \\ 0\,1\,0 \quad\;\; 0 \\ \quad\;\; \ddots \\ 0\,0\,0 \cdots 1 \\ 0\,0\,0 \cdots 1 \end{bmatrix}.$$

If we sample every other code symbol in $u = (u_1, u_2, \cdots, u_{384})$, the sampled code symbols $(u_1, u_3, \cdots, u_{383})$ should satisfy equation (9) since $(u_1, u_3, \cdots, u_{383})$ are the output symbols of the convolutional encoder. Mathematically, the sampling of every other symbols of $u = (u_1, u_2, \cdots, u_{384})$ can be described by the matrix multiplication uD_2, where

$$D_2 = \begin{bmatrix} 1\,0\,0 \cdots 0 \\ 0\,0\,0 \quad\;\; 0 \\ 0\,1\,0 \quad\;\; 0 \\ \;\vdots \quad \ddots \\ 0\,0\,0 \quad\;\; 1 \\ 0\,0\,0 \cdots 0 \end{bmatrix}.$$

Substituting v in (9) by uD_2, we have

$$uD_2[H]_{192} = 0. \tag{11}$$

Generally, if each code symbol in $u = (u_1, u_2, \cdots, u_{384})$ appears k times, where $k = 1, 2, 4, 8$, then the code symbol frame $u = (u_1, u_2, \cdots, u_{384})$ satisfies the following equations:

$$uE_k = 0, \tag{12}$$
$$uD_k[H]_{384/k} = 0, \tag{13}$$

where $E_1 = 0$, $D_1 = \mathbf{I}$, E_4, D_4, E_8, and D_8 can be derived in similar ways as E_2 and D_2.

3.3 Recovery of the long code sequence

To recover the long code sequence, the eavesdropper intercepts the downlink traffic channel, demodulates the intercepted data frames, and despreads the data frames using the Walsh code specific to the channel. Let $r = (r_1, r_2, \cdots, r_{384})$ denote the data frame after despreading. Correspondingly, let $c = (c(t_1), c(t_2), \cdots, c(t_{384}))$ denote the 384 bits of the long code sequence used for scrambling , $s = (s_1, s_2, \cdots, s_{384})$ denote the output of the block interleaver, and $u = (u_1, u_2, \cdots, u_{384})$ denote the input to the block interleaver. Figure 5 describes the relationship among these notations on the downlink traffic channel.

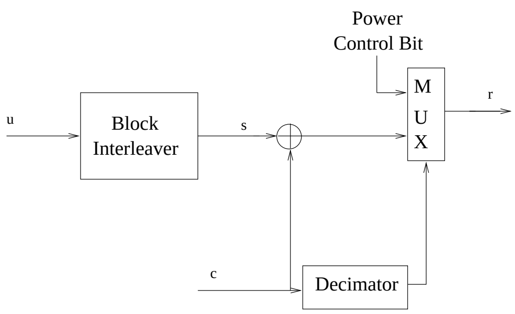

Fig. 5. Block interleaving and scrambling

The block interleaver is a permutation of the 384 input code symbols. The input symbols are entered as a 24×16 array and the interleaver produces a 24×16 output array. Table 1 describes the input array. The table is read down by columns from the left to the right. That is, the first input symbol u_1 is at the top left, the second input symbol u_2 is just below the first input symbol, and the

25th input symbol u_{25} is just to the right of the first input symbol. The output array is given by Table 2, which is read the same way as Table 1, that is, the first output symbol is u_1, the second output symbol is u_{65}, and the 25th output symbol is u_9.

Mathematically, the block interleaver can be represented by a permutation matrix P, namely,

$$s = uP, \tag{14}$$

where P is a 384×384 matrix, each row and column has only one 1.

For the moment, let's assume that the power control bits were not transmitted through the downlink traffic channel. Then we have

$$r = s \oplus c. \tag{15}$$

From (14) and (15),

$$u = (r \oplus c)P^{-1}. \tag{16}$$

Substituting (16) into (12) and (13),

$$(r \oplus c)P^{-1}E_k = 0, \tag{17}$$
$$(r \oplus c)P^{-1}D_k[H]_{384/k} = 0, \tag{18}$$

where k is equal to the number of times each code symbol appears in the frame.

Let $C(t_0) = (c(t_0), c(t_0 - 1), \cdots, c(t_0 - 41))$ denote the state of the LFSR that the eavesdropper would like to solve for. By (5), $c = (c(t_1), c(t_2), \cdots, c(t_{384}))$ can be represented as

$$c = C(t_0)\hat{A}. \tag{19}$$

where $\hat{A} = (A^{t_1 - t_0 - 1}, A^{t_2 - t_0 - 1}, \ldots, A^{t_{384} - t_0 - 1})a$.

Substituting (19) into (17) and (18), we have the following equations with $c(t_0), c(t_0 - 1), \cdots, c(t_0 - 41)$ as variables,

$$(r \oplus C(t_0)\hat{A})P^{-1}E_k = 0, \tag{20}$$
$$(r \oplus C(t_0)\hat{A})P^{-1}D_k[H]_{384/k} = 0. \tag{21}$$

Corresponding to every k, $k = 1, 2, 4, 8$, there are 200 or more linear equations involved in (20) and (21). Solving these linear equations, the state of the LFSR can be recovered.

Now, taking into account of the power control bits, many linear equations in (20) and (21) will no longer hold because of the corruption of the power control bits. The power control subchannel transmits at a rate of one bit every 1.25 ms. Using the puncturing technique, each power control bit which is two symbols long replaces two consecutive downlink traffic channel code symbols. Since the code symbols have a rate of 19.2 ksps, there will be one power control bit transmitted within every 24 code symbols. There are 16 possible starting positions for the power control bit. Each position corresponds to one of the first 16 code symbols within a 1.25 ms period. In each 1.25 ms period, a total of 24 bits from the long code sequence are used for scrambling. These bits are

1	25	49	73	97	121	145	169	193	217	241	165	289	313	337	361
2	26	50	74	98	122	246	170	194	218	242	266	290	314	338	362
3	27	51	75	99	123	147	171	195	219	243	267	291	315	339	363
4	28	52	76	100	124	148	172	196	220	244	268	292	316	340	364
5	29	53	77	101	125	149	173	197	221	245	269	293	317	341	365
6	30	54	78	102	126	150	174	198	222	246	270	294	318	342	366
7	31	55	79	103	127	151	175	199	223	247	271	295	319	343	367
8	32	56	80	104	128	152	176	200	224	248	272	296	320	344	368
9	33	57	81	105	129	153	177	201	225	249	273	297	321	345	369
10	34	58	81	106	130	154	178	202	226	250	274	298	322	346	370
11	35	59	83	107	131	155	179	203	227	251	275	299	323	347	371
12	36	60	84	108	132	156	180	204	228	252	276	300	324	348	372
13	37	61	85	109	133	157	181	205	229	253	277	301	325	349	373
14	38	62	86	110	134	158	182	206	230	254	278	302	326	350	374
15	39	63	87	11_1	135	159	183	207	231	255	279	303	327	351	375
16	40	64	88	112	136	160	184	208	232	256	280	304	328	352	376
17	41	65	89	113	137	161	185	209	233	257	281	305	329	353	377
18	42	66	90	114	138	162	186	210	234	258	282	306	330	354	378
19	43	67	91	115	139	163	187	211	235	259	283	307	331	355	379
20	44	68	92	116	140	164	188	212	236	260	284	308	332	356	380
21	45	69	93	117	141	165	189	213	237	261	285	309	333	357	381
22	46	70	94	118	142	166	190	214	238	262	286	310	334	358	382
23	47	71	95	119	143	167	191	215	239	263	287	311	335	359	383
24	48	72	96	120	144	168	192	216	240	264	288	312	336	360	384

Table 1: Downlink traffic channel interleaver input

1	9	5	13	3	11	7	15	2	10	6	14	4	12	8	16
65	73	69	77	67	75	71	79	66	74	70	78	68	76	72	80
129	137	133	141	131	139	135	143	130	138	134	142	132	140	136	144
193	201	197	205	195	203	199	207	194	202	198	206	196	204	200	208
257	265	261	269	259	267	263	271	258	266	262	270	260	268	264	272
321	329	325	333	323	331	327	335	322	330	326	334	324	332	328	336
33	41	37	45	35	43	39	47	34	42	38	46	36	44	40	48
97	105	101	109	99	107	103	111	98	106	102	110	100	108	104	112
161	169	165	173	163	171	167	175	162	170	166	174	164	172	168	176
225	233	229	237	227	235	231	239	226	234	230	238	228	236	232	240
289	297	293	301	291	299	295	303	290	298	294	302	292	300	296	304
353	361	357	365	355	363	359	367	354	362	358	366	356	364	360	368
17	25	21	29	19	27	23	31	18	26	22	30	20	28	24	32
81	89	85	93	83	91	87	95	82	90	86	94	84	92	88	96
145	153	149	157	147	155	151	159	146	154	150	158	148	156	152	160
209	217	213	221	211	219	215	223	210	218	214	222	212	220	216	224
273	281	277	285	275	283	279	287	274	282	278	286	276	284	280	288
337	345	341	349	339	347	343	351	338	346	342	350	340	348	344	352
49	57	53	61	51	59	55	63	50	58	54	62	52	60	56	64
113	121	117	125	115	123	119	127	114	122	118	126	116	124	120	128
177	185	181	189	179	187	183	191	178	186	182	190	180	188	184	192
241	249	245	253	243	251	247	255	242	250	246	254	244	252	248	256
305	313	309	317	307	315	311	319	306	314	310	318	308	316	312	320
369	377	373	381	371	379	375	383	370	378	374	382	372	380	376	384

Table 2: Downlink traffic channel interleaver output

numbered 0 through 23. The 4-bit binary number with values 0 through 15, formed from scrambling bits 23,22,21, and 20, are used to determine the position of the power control bit. Hence, within every 24 code symbols, the last 7 code symbols are not affected by the power control bits. These uncorrupted code symbols include $r_{18+24i}, r_{19+24i}, \ldots, r_{24+24i}, i = 0, 1, \cdots, 15$. Table 2 outlines the interleaver output uncorrupted by the power control bits. Those symbols in the dotted boxes in Table 2 can be recovered reliably from r and c. Correspondingly, those symbols in the dotted boxes in Table 1 can be recovered reliably from r and c. Code symbols in the dotted boxes in Table 1 can be divided into 7 groups. Each group contains 16 consecutive code symbols. The 7 groups are listed as follows:

49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256
305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320
337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352
369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384

In the following, we will extract those linear equations in (20) and (21) that are only related to the uncorrupted code symbols in r, such linear equations are called reliable linear equations.

(1) When the data rate is 19.2 ksps, we can not obtain any linear equations from (20) since code symbols are not repeated. In (21), there are 8 reliable linear equations. These reliable linear equations are related with code symbols in the 7th group, that is, s_{369} through s_{384}. Since the constraint length of the (2, 1, 8) convolutional encoder is 9, except for the first and the last 8 linear equations, other linear equations in (21) are related to 18 code symbols. From the structure of H, it is easy to find that the first 8 linear equations are related to the first 2, 4, 6, 8, 10, 12, 14, and 16 code symbols respectively. Similarly, the last 8 linear equations are related to the last 2, 4, 6, 8, 10, 12, 14, and 16 code symbols respectively.

(2) When the data rate is 9.6 ksps, there are 56 reliable linear equations in (20), which are related to the code symbols in the 7 groups. There are 4 reliable linear equations in (21), which are related to the code symbols in the 7th group. Hence, there are a total of 60 reliable linear equations in (20) and (21).

(3) When the data rate is 4.8 ksps, there are 86 reliable linear equations in (20) and 2 reliable linear equations in (21), with a total of 86 reliable linear equations in (20) and (21).

(4) When the data rate is 2.4 ksps, there are 98 reliable linear equations in (20) and 1 reliable linear equations in (21), with a total of 99 reliable linear equations in (20) and (21).

Hence, when the data rate is not 19.2 ksps, an eavesdropper can derive 60 or more reliable linear equations from one data frame. The eavesdropper can solve the reliable linear equations corresponding to the three data rates, 9.6

ksps, 4.8 ksps, and 2.4 ksps. If there are no solutions corresponding to any of the three data rates. The eavesdropper determines that the data rate should be 19.2 ksps. If there is a solution corresponding to one of the three data dates. The long code sequence can be computed and the positions of the power control bits can be determined. Also the corruption caused by the power control bits can be removed by the convolution code. The intercepted code symbols are tested against equations (17) and (18). If the test fails, the solution is incorrect. If there is only one solution passing the test, then long code sequence is found out. If more than one solution pass the tests, then use another frame to test every solution, this process continues until a unique solution is found out.

3.4 A robust attack

In the above attack, the eavesdropper needs to solve 4 sets of linear equations corresponding to the 4 data rates, 19.2 ksps, 9.6 ksps, 4.8 ksps, and 2.4 ksps. In the situation when more than one set of linear equations have solutions, the eavesdropper has to determine which solution is correct. This process can be avoided if the eavesdropper can construct a set of linear equations that hold for the 4 data rates. In the following, we will investigate reliable linear equations in (20) and (21) that hold for the 4 data rates.

When the data rate is not 19.2 ksps, the code symbols in u are repeated at least one time. As a result u_{2i} and $u_{2i+1}, 0 \leq i \leq 192$, must be equal, i.e.,

$$u_{2i} \oplus u_{2i+1} = 0, \quad 0 \leq i \leq 192. \tag{22}$$

When the data rate is 19.2 ksps, the code symbols in u are not repeated. But, from (13), we can get the following equation

$$u_{383} \oplus u_{384} = 0. \tag{23}$$

By (22), it can be concluded that (23) holds for the 4 data rates. From Table 2,

$$u_{383} = s_{192},$$

$$u_{384} = s_{384}.$$

Since r_{192} and r_{384} are not corrupted by the power control bits, by (15),

$$\begin{aligned} u_{383} &= r_{192} \oplus c(t_{192}), \\ u_{384} &= r_{384} \oplus c(t_{384}). \end{aligned}$$

Hence, it follows that

$$c(t_{192}) \oplus c(t_{384}) = r_{192} \oplus r_{384}. \tag{24}$$

Therefore, from every intercepted frame, the eavesdropper can always construct a reliable linear equation which is independent of the date rate associated with the frame. With 42 intercepted frames, the eavesdropper can construct a set of 42 reliable linear equations to solve for the long code sequence. Each frame lasts for 20 ms, 42 frames last for 840 ms, which is less than 1 second. Hence, by eavesdropping the downlink traffic channel 1 second, the eavesdropper can get enough information to recover the long code sequence. After recovering the long code sequence, the eavesdropper can despread the uplink traffic channel.

4 Concluding Remarks

The analysis of this paper demonstrates that IS-95 CDMA provides a lowwer than expected level of voice privacy. By eavesdropping the transmission on the downlink traffic channel for one second, an eavesdropper can recover the long code sequence used for voice privacy. The vulnerability of the voice privacy may have effect on the security of the authentication process since the long code mask is generated during the authentication process. As details of the authentication protocol are not publicly available due to export restriction, we will leave this problem for future research.

There are several reasons for the weak voice privacy. First, channel coding and symbol repetition leak information about the long code sequence, which is demonstrated by (12) and (13). Channel coding provides error correction capability by introducing redundancy in the transmitted information, while ciphers remove redundancy in the information. When error-correcting codes and ciphers are used in the same channel, the effects of error-correcting codes on ciphers should be carefully examined. Second, the block interleaver does not distribute power control bits in a frame uniformly. From Table 1, there are many consecutive code symbols which are not affected by the power control bits. Third, most importantly, the long code generator is not a good cipher. From (3), the long code sequence is a linear feedback shift register sequence. Linear feedback shift register sequence may be good for spread spectrum purpose but not good for cryptographic purpose. It had better design the two sequences separately, one used for spreading, and the other for voice privacy. It is difficult to design a sequence for both spread spectrum and voice privacy. There are different requirements for the two applications. Hence, a new cipher that provides strong voice privacy is required.

Last, we would like to emphasis that attacks described in this paper have not gone through any field test and we have no intention of performing any test in the future. To achieve a high level of voice privacy, the Telecommunication Industry Association (TIA) TR-45 has created a process for developing enhanced encryption algorithms for the next generation CDMA systems. The enhancement will use 128-bit private keys. Several algorithms, including an algorithm designed by the authors of the paper, have been submitted to TR-45 Ad Hoc Authentication Group (AHAG) for adoption of encryption standard for the next generation CDMA systems.

References

1. Ajay Dholakia, *Introduction to Convolution Codes and Applications*, Kluwer Academic Publishers, 1994.
2. Rajian Kuruppillai, Mahi Dontamsetti, Fil Cosentino, *Wireless PCS*, McGraw-Hill, 1997.
3. TIA/EIA/IS-95-A, Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System.