Hardening of Pin Numbers of Automated Teller Machines

Eltayeb Salih Abuelyman
Dean of the College of Computer and Information Sciences
Prince Sultan University, Riyadh 11586, Saudi Arabia

Summary
Automatic teller machines have given customers banking at leisure. Unfortunately such convenience is coming at a price that sometimes can be costly. While external ATM problems have been causing nightmare to both customers and the banking industry, vulnerabilities to ATM systems are enticing unethical insiders to cause customers damage from within. Recently, researchers detailed an elegant attack that yields a customer's personal identification number in as low as 15 tests. This paper demonstrates the attack in simpler terms and then suggests a hardening scheme for the vulnerability. The proposed hardening scheme is demonstrated at both the hardware and software level.

Keywords: Automated Teller Machines, Personal Identification Number, Decimalization, Hardening, Modulo arithmetic.

1. Introduction

Researchers have been concerned with solving security issues of Automated Teller Machines (ATM). Attempts have been made to eliminate, if not, reduce risks associated with the use thereof. This paper will first review alternatives to Personal Identification Numbers (pin). Such alternatives, according to their proponents, will have less probabilities of: theft; cracking; or shoulder surfing. The paper will also review a proposed use of the reverse of a pin number when an ATM customer is under threat by a robber. Finally, the paper will discuss published work that enables insiders to crack pin numbers and then propose hardening to counter such vulnerability. First, are there any alternatives to pin numbers?

1.1 Alternatives to pin numbers

Pin numbers are considered by many to be vulnerable because customers are likely to choose ones that are easy to guess or easily found written down somewhere. Pin numbers are also subject to cracking not only by outsiders, but also by bank insiders.

Furthermore, thieves are becoming bolder and creative in using skimmers to read the magnetic strip of ATM cards which contains account numbers, account types and banks numbers. To complete gathering of necessary information they use cameras to capture pins numbers. The thieves then make duplicate cards and use the pins to withdraw thousands of dollars from many accounts in a very short period of time directly from the bank ATM machine [1]. In search for answers, researcher proposed several alternatives and remedies.

Alternative to numeric pins include graphical passwords. Proponents of this alternative expect it to be easier to recall, less vulnerable and can have more symbols than their numeric counterparts [2]. Others use a series of clickable points on an image or require customer-specific series of lines drawn on a screen [3]. Yet another alternative is the use of icons. The authors claim that it is easier to remember a sequence of clickable icons. They require permutation of the icons every time a user accesses an ATM machine. The authors claim that in addition to the aforementioned advantages over the pin system, their approach will not suffer from over the shoulder surfing as do pin and graphical password systems [4]. Although some authors argue that graphical passwords are entered on a small screen with a reduced observation angle and they dismiss the likelihood of shoulder surfing [5]. The next subsection is about the protection of customers using their pins at ATMs.

1.2 Use of reverse pin for emergency handling

This subsection was taken from a report prepared in support of the case of Mr. Joe Zingher against the banking industry [6]. Several websites have posted articles related to the case [7-11]. { " Mr. Zingher proposed a protection system for customers of ATMs. His system is based on customers' Personal Identification Numbers (pin). He envisioned the system to report a robbery when a customer enters his/her pin number backward. Unfortunately the banking system rejected his proposal and left him with no option but to file a law suite. In 2005 Mr. Zingher asked me to become a witness in his law suite. Interestingly enough I found his idea to be feasible and works as follows: a card holder under robbery at an ATM machine should enter the reverse of his/her pin number. Such action would honor the transaction but at the same time reveal information about the location of the incident to the law enforcement offices. It would also activate additional cameras in the neighborhood. Mr. Zingher was told that his idea would require expensive physical reconfiguration of the ATM

hardware. Moreover, he was told that an international treaty forbids any changes to the existing ATM system"}. Despite the fact that the reverse pin idea is not directly related to the main theme of the paper, it leads to the analysis in the following subsection.

1.3 Usable pin numbers for reverse pin actions

To investigate the problem, it was logical to begin with the part that concerns banks the most: their losses. The banking industry will definitely be faced with a problem of reengineering the pin number system. Obviously there are many combinations of four decimal digit numbers the bank would have to retire. These are the combinations that are palindromes. That is, numbers of the form **XYYX** or **XXXX**. Out of the **10,000** possible four decimal digit numbers, there are **90** of the former and **10** of the latter. If we exclude these numbers we would be left with **9900** non-palindromes. Obviously half of these numbers must be eliminated to avoid having a customer with a pin equals **WXYZ** sending a false emergency signal ostensibly on behalf of another with pin number equals **ZYXW** or vice versa. Eliminating palindromes, a bank would be left with **4950** numbers that could be used. However, that is not the end of combinations banks have to retire. Our investigation revealed that there are banks that do not allow three sets of easy to remember numbers [12]. These sets are: the set of identical digits, as in **1111**; the set of sequences, as in **1234**; the set of numbers with two or more consecutive zeros. However, the first set has already been eliminated as part of the palindromes. The second and third sets would have **16** and **315** members respectively. Therefore we have to subtract **331** from **4950**. A bank would therefore have **4619** combinations for pin numbers. This number is definitely less that the number of ATM customers of an average bank in the United States. That implies the existence of multiple ATM cards with the same pin, something banks never disclose. Having given a quick overview, subsequent sections will concentrate on the pin number problems.

The rest of the paper is organized as follows. In section 2 the calculation of pin numbers is reviewed. This is followed by protection of the pin number support systems in section 3. Section 4 will present implementation of a permutation function for hardening against the system's vulnerabilities. Insider's attempts to crack pins are presented in section 5. Finally, the conclusion is given in section 6.

2. Calculation of pin numbers

A customer's pin number is calculated from his/her bank account number. First, the account number is encrypted using the Digital Encryption Standard (DES). The ASCII output of DES is then decimalized, i.e., converted into a hexadecimal number. Finally, the four most significant digits of the resulting hexadecimal number are extracted and used as the pin number. The next subsection will address decimalization.

2.1 Decimalization

Since hexadecimal digits include the decimal digits **0** through **9** and the alphabetical letters **A** through **F**, a conversion of the letters was necessary. To that end, a decimalization table was established. The following is an example showing step by step computation of a pin from a randomly selected integer **N**. First, let us discuss the value of **N** chosen for this study.

There is a probability of **1** out of $\mathbf{10^{16}}$ that **N** is equal to an actual ATM card number. We will not disclose our version of the DES cipher of **N** albeit the chances are slim for it to be identical to a real card identifier. Besides, we also altered the cipher to further minimize the likelihood that both **N** and its pin match a real ATM pair. The next step is to show our randomly selected **N** and the hexadecimal number resulting from conversion of the altered cipher of **N** where:

1. **N = 5432 1254 3405 3879**
2. **Hex (DES[N]) = 9B5C 290A 6007 51CD**

The most significant four digits of the hexadecimal number in (2) are referred to as a prefix. The second and fourth digits of the prefix are a "**B**" and a "**C**" respectively. However, alphabetic letters are unavailable on any ATM keypad for practical reasons. Such keypads come with the decimal digits only. This is the justification behind use of a decimalization table for mapping the letters **A** through **F** to decimal digits. Table 1 shows the Typical Decimalization (Typ-Decim) of hexadecimal digits of the first row on the second. For our example, the prefix **9B5C** would thereby be decimalized to a pin equals **9152**. All tables are at the end of the paper for convenience of the presentation.

Practically speaking, not all banks use the same technique for computing a pin. Some assign unchangeable pin numbers to their customers. Others give customers chances to enter their own. However, what a customer enters is not the actual pin used by the ATM system. To save time for the reader, we will simplify the process by saying that the system firstly computes a pin for the customer. Secondly, it subtracts the customer's chosen pin from the one generated by the system and stores the difference as offset for future

references. When a customer enters his/her chosen pin at any ATM, the offset is added to it and verification takes place.

For example, let us assume that a customer entered his/her birth year **1984** for a pin. Assuming **N** to be the account number, the system will generate, as described before, a pin equals to say **9152**. The customer's chosen pin (**1984)** is subtracted from the system's pin (**9152**) and the difference **(7168**) is stored as the future offset. The next subsection will address pin hacking.

2.2 Pin Hacking: Brute force approach

Without retiring any number, determination of the right pin will take up to **10000** guesses. On the average, the same would take up to **5000** guesses. If we retire the combinations discussed in section 1.3, we would be left with **4619** usable numbers. In this case a person would need up to **4619** tries to determine a pin number. Equally, the average number of tries to determine a pin would be **2310**. Brute force determination of a pin is therefore improbable. What makes the situation even worse is that one gets only three trials before the card is locked up by the ATM machine. That is, one has a probability of **1** out of **770** to correctly guess a pin. Hence, the system is not as vulnerable from outside as it is from inside. Mike Bond and Piotr Zielinski gave a more efficient alternative for bank insiders [13]. In the next section we will discuss their decimalization table's approach which uses 15 tries to determine a pin.

2.3 Pin Cracking: Digit Decimalization table approach

In reference [13] they used the typical decimalization table to orchestrate a process for discovering a pin. Basically, they assume that an attacker has a Digit Decimalization Table (**DDT**) for each decimal digit. For a decimal digit **d**, the **DDT** table contains **1's** in columns where **d** appears as their header and 0's otherwise. For example, the **DDT** for the numbers **7,** herein referred to as $\mathbf{D_7}$, is shown on table 2.

The choice of the number **7** for the **DDT** is taken from reference [13]. According to the reference, an attacker can use a Trial Pin (**TP**) 0000 against $\mathbf{D_7}$ to check for the existence of the number **7** in the Pin Under Attach (**PUA**). It is clear, according to the authors, that the number **7** is not part of the **PUA**. Hence, with one **TP** the attacker would be able to eliminate the number **7**. The process continues until the attacker is able to eliminate all but the numbers that are part of the **PUA**. The attacker's next step is to determine the correct order of the digits. The complete process won't be further discussed in this paper. Interested readers are referred to the original document for more information.

I have to admit my struggle to understand the argument used by the authors for retiring the number **7**. This may be a deliberate deception of readers by the authors to avoid explaining the attack via a realistic decimalization table. Nevertheless, the authors gave enough clues that I will use to discuss a vulnerability that would lead to pin cracking and then I will propose a method of hardening it.

2.4 Pin cracking via decimalization table

The vulnerability discussed in this paper which may lead to cracking a pin depends on two key statements made by the authors. These statements are as follows:

a) A decimalization table is many-to-one mapping between the hexadecimal digits and the numeric digits.
b) A decimalization table should not be changed without permission.

On the typical decimalization table, every decimal digits is listed once except for the digits **0** through **5** which are listed twice each. However, for this section I will use table 3 instead of the typical decimalization on table 1.

Let us assume that an insider can change the decimalization table. To clarify the process, a **DDT** for all the numeric digits is presented on table 4.

Without loss of generality, let us assume that an insider is to crack a **PUA** that is equal to **8359.** To check for the existence of the decimal digit **0** in the **PUA,** the insider would add row $\mathbf{D_0}$ from table 4 to the Alt-Decim row of table 3 and store the result on a test table as shown on table 5. The only difference between Table 3 and the test table is the decimalization of the digit **9,** which changed from **0** to **1**. Table 5 will definitely fail to decimalize any digit to **0**.

Authentication of the **PUA** which is equal to **8359** would not be affected because it doesn't need to decimalize a digit to **0**. The insider would therefore conclude that the digit **0** is not part of the **PUA**. In a similar manner, one can easily see that generating test tables for $\mathbf{D_1}$ and $\mathbf{D_2}$ and testing for the digits **1** and **2** respectively would reveal that neither would have any effect on the authentication of **PUA**. This leads the attacker to believe that the numeric digits **1** and **2** are not part of the **PUA**. However, when the insider generates a test table for $\mathbf{D_3}$ as shown on table 6, the conclusion would be different.

This time the authentication of the PUA would fail

because no digit would decimalize to **3**. The insider would then conclude that the number **3** is part of the **PUA**. The process would continue in a similar manner until all the digits that are not part of the **PUA** are eliminated. At this point the insider would have put one foot at the door. The next step is to figure out the order of the digits, which is discussed in reference [13].

One way to guard against cracking pin numbers via the decimalization table is to ensure that a single person is not allowed to make changes to the table according to the authors. Although this measure is reasonable, it may not be guaranteed all the time. A solution to this problem is proposed in the following section.

3. Protection of the decimalization table

What concerns us here is that the typical decimalization table, or any alternative to it, should not be stored in their original form. Encryption was suggested by the authors to address such a vulnerability. However, a bijective function is proposed in this document to permute the hexadecimal entries of the table for hardening the vulnerability. Entries should be permuted in a way that even an insider would not be aware of. The use of a function **f** that maps each of the Hex-Dig digits to a unique hexadecimal digit is suggested for that purpose.

Our proposed solution will depend on the fact that all the hexadecimal digits are members of the set of residues modulo 16. Henceforth, we will use properties of the set of residues modulo 16 to find a non-trivial permutation of these residues as discussed in the nest subsection.

3.1 Permutation of the hexadecimal digits

Let us denote the set of integers modulo **16** by $\mathbf{Z_{16}}$ and the required permutation function by **f** where **f** maps its domain $\mathbf{Z_{16}}$ to its range $\mathbf{Z_{16}}$ which is denoted as follows.

$$\mathbf{f: Z_{16} \rightarrow Z_{16}}$$

For **f** to map a digit to its inverse the set $\mathbf{Z_{16}}$ must be a mathematical field. However, for $\mathbf{Z_{16}}$ to be a field the function **f** must map each element in $\mathbf{Z_{16}}$ to a unique inverse in a one to one onto manner.

The function **f** is a one to one onto function defined as follows: for any element **p** in $\mathbf{Z_{16}}$ there exists an element **C** in $\mathbf{Z_{16}}$ such that **f(p)** is equal to **C** where **f(p)** is given by:

$$\mathbf{f(p) = 3^* p \text{ MOD } 16 - 2^*(p \text{ MOD } 2)} \qquad \mathbf{p < 16} \quad \mathbf{(1)}$$

To prove that **C** is the inverse of **p**, and vice versa, we need to show that:

$$\mathbf{f(C) = f\{ f(p)\} = p} \qquad \mathbf{(2)}$$

To show that $\mathbf{p = f\{ f(p)\} = f(C)}$, the proof will be broken into two cases: the first is when **p** is equal to **e** where **e** is an even integer. The second is when **p** is equal to an odd integer **o**. However, before we proceed with the proof we need to remind the reader that in modulo arithmetic and for any mathematical operation **#** and integers **a**, **b** and **Z**, the following is valid:

$$(\mathbf{a \# b}) \text{ MOD } Z = \{ \mathbf{a} \text{ MOD } \mathbf{Z} \# \mathbf{b} \text{ MOD } \mathbf{Z}\} \text{MOD } \mathbf{Z}$$

Case 1: p = e

Given **e** an even integer, equation 1 can be reduced to the following:

$$\mathbf{f(e) =} 3 \mathbf{^* e} \text{ MOD } 16 \qquad \mathbf{(3)}$$

Thus, we need to show that: $\mathbf{f(C) = e}$ given $\mathbf{f(e) = C}$

Since in general $f(\mathbf{C}) = f\{\mathbf{f(p)}\}$, and $\mathbf{p = e}$, it follows that:

$$\begin{aligned} f(\mathbf{C}) &= f\{\mathbf{f(e)}\} \\ &= \mathbf{3}^*\{f(e)\} \ \mathbf{MOD\ 16} \\ &= \mathbf{3}\{3^* e \text{ MOD } 16\} \ \mathbf{MOD\ 16} \\ &= \mathbf{9^* e \ MOD\ 16} \\ &= \mathbf{8^* e \ MOD\ 16 + e \ MOD\ 16} \\ &= \mathbf{e} \end{aligned}$$

Case 2: m = o

Given **o** an odd integer, equation 1 can be rewritten as follows:

$$\mathbf{f(o) =} 3^* \mathbf{o} \text{ MOD } 16 - 2 (\mathbf{o} \text{ MOD } 2)$$

Thus, we need to show that: $\mathbf{f(C) = o}$ given $\mathbf{f(o) = C}$

Since in general $f(\mathbf{C}) = f\{\mathbf{f(p)}\}$, and $\mathbf{p = o}$, it follows that:

$$\begin{aligned} f(\mathbf{C}) &= f\{\mathbf{f(o)}\} \\ &= \mathbf{3}[3^* o \text{ MOD } 16 - 2^*(o \text{ MOD } 2)] \mathbf{MOD\ 16} \\ &\quad - \mathbf{2}^*[3 o \text{ MOD } 16 - 2(o \text{ MOD } 2)] \mathbf{MOD\ 2} \\ &= \mathbf{9^* o \ MOD\ 16 - 6\ (o \ MOD\ 2) - 2[3\ o \ MOD\ 2]} \\ &= \mathbf{9^* o \ MOD\ 16 - 6 - 2} \\ &= \mathbf{o \ MOD\ 16 + 8\ o \ MOD\ 16 - 8} \\ &= \mathbf{o + 8 \ MOD\ 16 - 8} \\ &= \mathbf{o} \end{aligned}$$

To guarantee that inverses are unique one needs to show that f(**w**) is equal to f(**y**) if, and only if, **w** is equal to **y.** This condition is trivial in modulo arithmetic since both **w** and **y** belong to the field Z_{16}.

The hexadecimal digits would then be permuted by the function **f** as shown on table 7 below. The proposed solution will harden all but pins that are formed from the digits **0**, **1**, **8**, and **9.** Since there are 4! or 24 different permutations of four digits, it follows that the hardening

would cover more than 99% of the pin numbers. This solution can easily be realized via software. In the next section, a possible scenario for hardware implementation is presented.

4. Implementation of the permutation function

Logically, the next step is to suggest a hardware implementation for the function. However, at this point, only the Boolean equations that can be used to derive **C** from **p** or vice versa are discussed. In a follow up paper, the complete details will be provided.

To demonstrate the results and without loss of generality, the number **13** ($\mathbf{D_{16}}$) is our **p**. From equation 1 we have:

$\mathbf{C}$ = f(**p**)
= f(13)
= 3 * 13 MOD 16 – 2(13 MOD 2)
= 39 MOD 16 – 2
= 5.

Conversely: from equation 2 we can show that

p = f(**c**)
= 3 *5 MOD 16 – 2(5 MOD 2)
= 15 MOD 16 – 2
= 13.

Let us assume that **C** and **p** are expressed in binary as $C_1C_2C_3C_4$ and $p_1p_2p_3p_4$ respectively. Furthermore, let us assume that for any digit **d**, **d'** represent the complement of **d** then:

$\mathbf{C_1} = \mathbf{p'_1p_2p'_3} + \mathbf{p_1p'_2} + \mathbf{p_2p_3}$ (4)
$\mathbf{C_2} = \mathbf{p_2 @ p_3}$ (5)
$\mathbf{C_k} = \mathbf{p_k}$ for k = 3,4 (6)

Where the **@** character is used to represent the **exclusive-or** operator.

Now from the implementation equations 4, 5, and 6 we can find the value of **C** when $\mathbf{p} = \mathbf{13_{10}} = \mathbf{1101_2}$ as follows:

$\mathbf{C_1} = \mathbf{p'_1*p_2*p'_3} + \mathbf{p_1*p'_2} + \mathbf{p_2*p_3} = 0 + 0 + 0 = \mathbf{0}$
$\mathbf{C_2} = \mathbf{p_2 @ p_3} = 1 \mathbf{@} 0 = \mathbf{1}$
$\mathbf{C_3} = \mathbf{0}$
$\mathbf{C_4} = \mathbf{1}$

We can also go back to find **P** from $\mathbf{C} = \mathbf{0101_2}$ as follows:

$\mathbf{P_1} = \mathbf{C'_1*C_2*C'_3} + \mathbf{C_1*C'_2} + \mathbf{C_2*C_3}$
$= 1 + 0 + 0$
$= \mathbf{1}$
$\mathbf{P_2} = \mathbf{C_2 @ C_3}$
$= 1 \mathbf{@} 0$
$= \mathbf{1}$
$\mathbf{P_3} = \mathbf{0}$
$\mathbf{P_4} = \mathbf{1}$

This shows that $\mathbf{P} = \mathbf{1101_2}$

5. Detection of insider's attempts

The function **f** was used to compute entries of the third row from the firs and the results are shown on table 8.. An insider would be familiar with the first two rows of the table. However, he/she would not see rows three and four. The hardened decimalization would use the first, third and fourth rows of the table. Further information about the implementation and improvement of the process to harden against all possible combinations will be given in a future paper.

6. Conclusion

This paper proposed a hardening against a systematic attack that renders Personal Identification Numbers of ATM customers to interested bank insiders. The attacker manipulates stored data in a way that enables him/her via the process of elimination to determine the four digits of a pin number. The attacker then uses other techniques to find the correct order of the digits. The proposal suggests encoding of vulnerable data in a simple but hard to detect way, thereby complicating the cracking process. The encoding will provide protection to 99% of the space of pin numbers.

References

[1] http://www.uboc.com/about/main/0,,2485_703976951,00.html
[2] Bogdan Hoanca and Kenrick Mock "Eye Tracking Research & Application" Proceedings of the 2006 symposium on Eye tracking research & applications San Diego, California , pp 35 – 35, 2006
[3] Davis, D., Monrose, F. and Reiter, M. "On User Choice in Graphical Password Schemes." Proceedings of the 13th USENIX Security Symposium, San Diego, CA, USA 2004.
[4] Bogdan Hoanca, Kenrick J. Mock "Screen oriented technique for reducing the incidence of shoulder surfing". Proceedings of the Security and Management, pp 334-340, Las Vegas 2005:
[5] Jansen, W. " Authenticating Mobile Device Users through Image Selection." The Internet Society: Advances in Learning, Commerce and Security, v30, p 2004
[6] Abuelyaman E. " Effectiveness of Reverse ATM PIN for Protection" Unpublished Technical Report, Illinois State University, 2005
[7]http://sandwalk.blogspot.com/2007/01/reverse-pin-at-atm-summons-police.html#comment-4395864760311174591
[8]http://sandwalk.blogspot.com/2007/03/more-on-reverse-pin-numbers.html
[9] http://www.snopes.com/business/bank/pinalert.asp

[10]http://viewsreviews.wordpress.com/2006/11/24/atm-reverse-pin/
[11] http://www.hoax-slayer.com/reverse-pin-ATM.shtml
[12] http://en.wikipedia.org/wiki/Personal_identification_number
[13] Mike Bond and Piotre Zielinski "Decimalization Table attack for Pin Cracking" A Technical Report, Computer Laboratory, University of Cambridge UCAM-CL-TR 560

Eltayeb Salih Abuelyaman received a PhD degree in Computer Engineering from the University of Arizona in 1988. He served as faculty member at various universities in the US for 18 years before moving to Prince Sultan University in Saudi Arabia where he served as a Faculty Member and Director of the Information Technology and Computing Services. Currently, he serves as the Dean of the College of Computer and Information Sciences. His current research Interest is in the areas of Computer Networks and Information Security and Database.

Hex-Digits	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Typ-Decim	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5

Table 1. Decimalization of Hexadecimal numbers

Typ-Decim	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5
D_7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Table 2. Decimalization table for the number 7.

Hex-Digits	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Alt-Decim	6	3	8	5	9	1	4	3	2	0	6	8	5	9	4	7

Table 3. Altered Decimalization Table

Alt-Decim	6	3	8	5	9	1	4	3	2	0	6	8	5	9	4	7
$\mathbf{D_0}$	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
$\mathbf{D_1}$	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
$\mathbf{D_2}$	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
$\mathbf{D_3}$	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
$\mathbf{D_4}$	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
$\mathbf{D_5}$	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
$\mathbf{D_6}$	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
$\mathbf{D_7}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
$\mathbf{D_8}$	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
$\mathbf{D_9}$	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

Table 4. Digit decimalization table for the numeric digits

Hex-Digits	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Alt-Decim	6	3	8	5	9	1	4	3	2	*1*	6	8	5	9	4	7

Table 5. Test table for D_0

Hex-Digits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
Alt-Decim	6	*4*	8	5	9	1	4	*4*	2	0	6	8	5	9	4	7

Table 6 Test table for D_3

Hex-Digits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
Perm-Hex	0	1	6	7	C	D	2	3	8	9	E	F	4	5	A	B

Table 7. Permutation of the Hexadecimal digits

Hex-Digits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
Typ-Decim	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5
Perm-Hex	**0**	**1**	**6**	**7**	**C**	**D**	**2**	**3**	**8**	**9**	**E**	**F**	**4**	**5**	**A**	**B**
Per-Decim	0	1	6	7	2	3	2	3	8	9	4	5	4	5	0	1

Table 8. Permuted Decimalization Table