CRYPTANALYSIS OF ADFGVX ENCIPHERMENT SYSTEMS

Alan G. Konheim

Computer Science Department

University of California

Santa Barbara, California 93106 USA

Extended Abstract

The ADFGX cryptographic system, invented by Fritz Nebel, was introduced by Germany during World War I on March 5, 1918. The names ADFGX and ADFGVX for the successor system refer to the use of only five (and later six) letters A, D, F, G, (V,) X in the ciphertext alphabet. Kahn [KA] suggests that these letters were chosen because differences in Morse International symbols .

A	•-	D	-••	F	••-•
G	--•	V	•••-	X	-••-

aided the prevent misidentification due to transmission noise.

The ADFGVX system is historically important since it combined both letter substitution and fractionation (transposition). Although Allied cryptanalysts did not develop a general method for the solution of ADFGVX ciphertext, Georges Painvin of the French Military Cryptographic Bureau found solutions which significantly effected the military outcome in 1918. This paper proposes a new method for the cryptanalysis of ADFGVX-type systems.

Let **A** denote an alphabet of $m = M^2$ "letters" which we henceforth identify with the set of integers $\mathbf{Z}_m = \{0, 1, \ldots, m-1\}$. The ADFGVX key (SUB, π) has two components; the first, an M by M array SUB containing an arrangement of the letters of $\mathbf{Z}_m$. For example, with $m = 25$

$$\text{SUB} = \left| \begin{array}{ccccc} C & R & Y & P & T \\ O & G & A & H & B \\ \mathrm{D} & \mathrm{E} & \mathrm{F} & \mathrm{I} & K \\ L & M & N & Q & S \\ U & V & W & X & Z \end{array} \right|$$

The second is a transposition

$$\pi = (\pi(0), \pi(1), \ldots, \pi(N-1))$$

on N places.

The steps in an ADFGVX encipherment are as follows:

ADFGVX(1) : Plaintext of n *m-letters*

$$\overline{x} = (x_0 , x_1 , \ldots , x_{n-1}) \qquad x_i \in \mathbf{Z}_m$$

is expanded into a 2n-gram of *M-letters*

$$\overline{z} = (z_0 , z_1 , \ldots , z_{2n-1}) \qquad (z_{2i} , z_{2i+1}) = (x_{i,0} , x_{i,1}) \qquad x_{i,j} \in \mathbf{Z}_M$$

The $\{ z_i \}$ are determined by the substitution *SUB*

$$x_i \quad \rightarrow \quad (x_{i,0} , x_{i,1}) \qquad x_{i,0} , x_{i,1} \in \mathbf{Z}_M$$

where $x_{i,0}$ and $x_{i,1}$ are the row and columns coordinates of x_i in *SUB*.

ADFGVX(2) : The "expanded" plaintext $\overline{z}$ is arranged in a (possibly) "ragged" *z-array* containing r rows of N columns and a (possible) $(r + 1)^{st}$ "short" row of $s < N$ columns;

$$2n = rN + s \qquad (0 \leq s < N)$$

$$z : \left| \begin{array}{ccccc} z_0 & z_1 & \ldots & \ldots & z_{N-1} \\ z_N & z_{N+1} & \ldots & \ldots & z_{2N-1} \\ . & . & \ldots & \ldots & . \\ . & . & \ldots & \ldots & . \\ . & . & \ldots & \ldots & . \\ z_{(r-1)N} & z_{(r-1)N+1} & \ldots & \ldots & z_{rN-1} \\ z_{rN} & \ldots & z_{rN+s} & & \end{array} \right|$$

ADFGVX(3) : The ciphertext $\overline{y} = (y_0, y_1, \ldots, y_{2n-1})$ is the concatentation of the columns of the z-array in the order defined by π.

We assume the length N of the transposition π is known, although the method will suggest a procedure to test a value N as a presumptive transposition length. The ciphertext

$$\overline{y} = (y_0 , y_1 , \ldots , y_{2n-1}) \qquad 2n = rN + s$$

is the concatention of segments $\{ \overline{y}^{(i)} \}$ of $\overline{y}$ which correspond to the entries in a single column of the z-array. We call $\overline{y}^{(i)}$ a ***column vector***. The cryptanalysis will follow these steps:

Step 1 : Determine which column vectors $\{ \overline{y}^{(i)} \}$ are adjacent in the z-array.

Step 2 : Determine the relative order of the pair $\overline{y}^{(\alpha_i)}$ $\overline{y}^{(\beta_i)}$ of adjacent column vectors

$$\overline{y}^{(\alpha_i)} \, \overline{y}^{(\beta_i)} \quad or \quad \overline{y}^{(\beta_i)} \, \overline{y}^{(\alpha_i)}$$

Step 3 : Recover the substitution *SUB*.

Step 4 : Recover the transposition π.

To carry out *Step 1,* we detect the "dependence" between the marginal "letter counts" $N_s^{(i)}$, $N_t^{(j)}$ and $N_{s,t}^{(i,j)}$ for a pair of column vector $\vec{y}^{(i)}$ $\vec{y}^{(j)}$ where

$$N_s^{(i)} = \sum_{t=0}^{M-1} N_{s,t}^{(i,j)} \qquad N_t^{(j)} = \sum_{s=0}^{M-1} N_{s,t}^{(i,j)}$$

and $N_{s,t}^{(i,j)}$ is equal to the number of solutions $k = 0 , 1 , \ldots$ of

$$y_{ir+k} = s \qquad y_{jr+k} = t \qquad 0 \leq s , t < M$$

Dependence will be detected by a variant of the χ^2*−test.*

Having identified and ordered (*Step 2*) adjacent column vectors $\vec{y}^{(\alpha_i)}$, $\vec{y}^{(\beta_i)}$, the sum

$$N_{s,t} = \sum_i N_{s,t}^{(\alpha_i , \beta_i)}$$

is the count of m-letters $(s , t) \in \mathbf{Z}_M \times \mathbf{Z}_M = \mathbf{Z}_m$ characteristic of a monalphabetic substitution. *SUB* may then be recovered by standard techniques. Having removed the effect of the substitution, the arrangement of the column vector pairs $\{ (\vec{y}^{(\alpha_i)} , \vec{y}^{(\beta_i)}) \}$ to reconstitute the z-array requires the solution of a pure transposition system.

The analysis requires an examination of several cases:

Case 1 :	$N \equiv 0\ (modulo\ 2)$	$s = 0$
Case 2 :	$N \equiv 0\ (modulo\ 2)$	$0 < s < N$
Case 3 :	$N \equiv 1\ (modulo\ 2)$	$s = 0$
Case 4 :	$N \equiv 1\ (modulo\ 2)$	$0 < s < N$

Details and proofs will appear in a paper submitted to the *IEEE Transactions on Information Theory.*