Chemical Combinatorial Attacks on Keyboards

Eric Brier
Gemplus Card International
Applied Research & Security Centre
La Vigie. Avenue des Jujubiers
La Ciotat, F-13705, France
eric.brier@gemplus.com

David Naccache, Pascal Paillier
Gemplus Card International
Applied Research & Security Centre
34 rue Guynemer
Issy les Moulineaux CEDEX, F-92447, France
{david.naccache,pascal.paillier}@gemplus.com

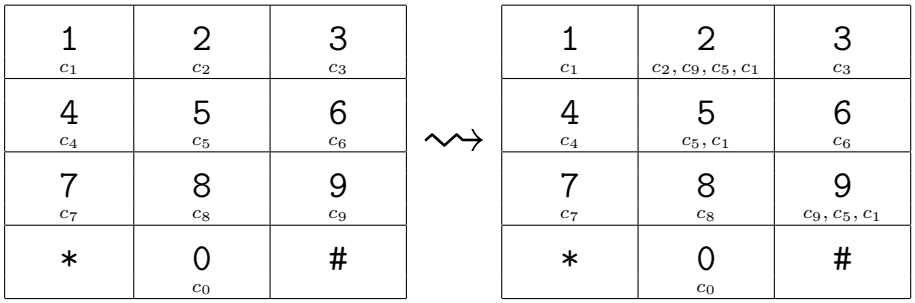
Abstract. This paper presents a new attack on keyboards.
The attack consists in depositing on each keyboard key a small ionic salt quantity (*e.g.* some NaCl on key 0, some KCl on key 1, LiCl on key 2, $SrCl_2$ on key 3, $BaCl_2$ on key 4, $CaCl_2$ on key 5...). As the user enters his PIN, salts get mixed and leave the keyboard in a state that leaks secret information. Nicely enough, evaluating the entropy loss due to the chemical trace turns out to be a very interesting combinatorial exercise.
Under the assumption that mass spectroscopic analysis can reveal with accuracy the mixture of chemical compounds generated by the user, we show that, for moderate-size decimal PINs, the attack would generally disclose the PIN.
The attack may apply to door PIN codes, phone numbers dialed from a hotel rooms, computer keyboards or even ATMs.
While we did not implement the chemical part of the attack, a number of mass spectrometry specialists confirmed to the authors its feasibility.

1 Introduction

This paper presents a new attack on keyboards and PIN-pads.

The attack consists in depositing on each keyboard key a small ionic salt quantity (*e.g.* some NaCl on key 0, some KCl on key 1, LiCl on key 2, $SrCl_2$ on key 3, $BaCl_2$ on key 4, $CaCl_2$ on key 5...). As the user enters his PIN, salts get mixed and leave the keyboard in a state that leaks secret information.

This first phase of the attack is illustrated below for the PIN `1592`.



The second part of the attack consists in collecting samples from the keyboard and analyzing these using a mass spectrometer (*e.g.* [1]).

In mass spectrometry, a substance is bombarded with an electron beam having sufficient energy to fragment the molecule. The positive fragments which are produced (cations and radical cations) are accelerated in a vacuum through a magnetic field and are sorted on the basis of mass-to-charge ratio. Since the bulk of the ions produced in the mass spectrometer carry a unit positive charge, the value m/e is equivalent to the molecular weight of the fragment. The analysis of mass spectroscopy information involves the re-assembling of fragments, working backwards to generate the original molecule. A schematic representation of a mass spectrometer is shown below:

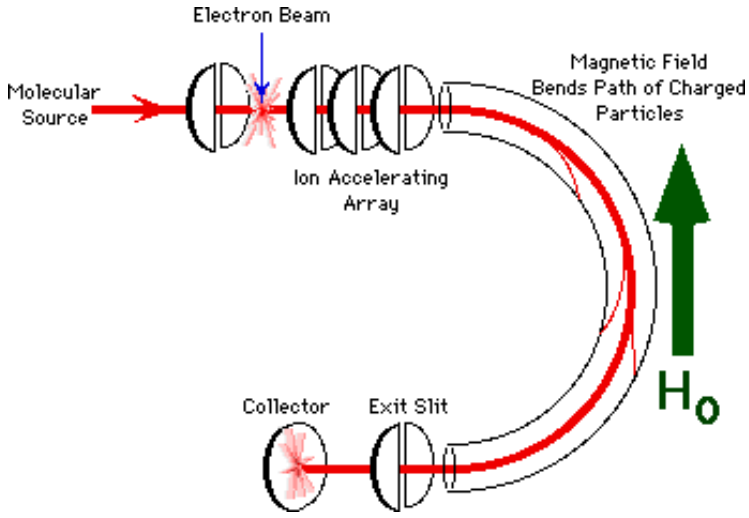


Figure A.

A very low concentration of sample molecules is allowed to leak into the ionization chamber (which is under a very high vacuum) where they are bombarded by a high-energy electron beam. The molecules fragment and the positive ions produced are accelerated through a charged array into an analyzing tube. The path of the charged molecules is bent by an applied magnetic field. Ions having low mass (low momentum) will be deflected most by this field and will collide with the walls of the analyzer. Likewise, high momentum ions will not be deflected enough and will also collide with the analyzer wall. Ions having the proper mass-to-charge ratio, however, will follow the path of the analyzer, exit through the slit and collide with the Collector. This generates an electric current, which is then amplified and detected. By varying the strength of the magnetic field, the mass-to-charge ratio which is analyzed can be continuously varied.

The output of the mass spectrometer shows a plot of relative intensity *versus* the mass-to-charge ratio (m/e). The most intense peak in the spectrum is termed the *base peak* and all others are reported relative to it's intensity. The peaks

themselves are typically very sharp, and are often simply displayed by the device as vertical lines.

The process of fragmentation follows simple and predictable chemical pathways and the ions which are formed will reflect the most stable cations and radical cations which that molecule can form. The highest molecular weight peak observed in a spectrum will typically represent the parent molecule, minus an electron, and is termed the *molecular ion*.

Having inferred what chemicals each keys contain, the attacker can proceed and try the PIN candidates one by one. The next section focuses on this third combinatorial aspect of the attack[1].

2 Combinatorial Analysis

We denote by $\mathcal{P}_\ell^d$ the set of PINs of length ℓ chosen amongst d digits. The chemical trace of a PIN is a map which associates to each digit the set of its predecessors on the keyboard. We denote by $\tau(p)$ the chemical trace of PIN p and define the set of all possible traces as $\mathcal{T}_\ell^d = \tau\left(\mathcal{P}_\ell^d\right)$.

2.1 Action of Permutations

The permutation group $\mathcal{S}_d$ on digits has a natural action on PIN values and this action extends to the traces. We define the cosets under this action as:

$$\widetilde{\mathcal{P}_\ell^d} = \mathcal{P}_\ell^d/\mathcal{S}_d \quad \text{and} \quad \widetilde{\mathcal{T}_\ell^d} = \mathcal{T}_\ell^d/\mathcal{S}_d$$

The trace map extends to cosets as a map:

$$\tilde{\tau} : \widetilde{\mathcal{P}_\ell^d} \longrightarrow \widetilde{\mathcal{T}_\ell^d}$$

Representatives of cosets in $\widetilde{\mathcal{P}_\ell^d}$ are easy to define: these are PINs wherein the digit `0` is used before `1`, which is used before `2` *etc.* We call such PINs *canonical PINs*.

The cardinality of $\widetilde{\mathcal{P}_\ell^d}$, which equals the number of canonical PINs of length ℓ, is easy to compute by virtue of the following proposition:

Proposition 1. *$\sharp\widetilde{\mathcal{P}_\ell^d}$ is the exponential Bell number $\mathcal{B}_\ell$ as soon as $d \geq \ell$.*

Proof. It suffices to exhibit a bijective map between canonical PINs and partitions of the set $\{1, 2, \cdots, \ell\}$. To associate a partition to a canonical PIN, we pack together positions where the digits take the same value. To map back a partition to a canonical PIN, we just associate a value to each partition, such that new digits occur in ascending order. □

[1] It should be stressed that while we did not experiment the chemical part of the attack, a number of spectrometry experts (Henri Boccia, Jorge Davilla *etc.*) confirmed to the authors its practical feasibility.

The following definitions will be useful in order to study the set $\widetilde{\mathcal{T}_\ell^d}$.

Definition 1. *Let p be a PIN. The signature of a digit δ in p is given by $(a, b) \in \mathbb{N} \times \mathbb{N}$, where a is the number of predecessors of δ and b the number of its successors.*

In the following definition, we use an ordering of the set $\mathbb{N} \times \mathbb{N}$. The chosen ordering is not relevant, the lexicographic order being just fine for our purpose.

Definition 2. *Let p be a PIN. The signature of p is the ordered list of the signatures of p's digits.*

Example: The signature of $p =$ `47524` is $\{(2,4),(3,3),(4,2),(4,4)\}$. This signature was computed as follows: Digit `7` has *two* predecessors (`4` and itself) and *four* successors (itself, `5`, `2`, and `4`) hence the element $(2,4)$ in the signature. Digit `5` has *three* predecessors (`4`, `7` and itself) and *three* successors (itself, `2` and `4`) hence the element $(3,3)$ in the signature. Digit `2` has *four* predecessors (`4`, `7`, `5` and itself) and *two* successors (itself and `4`) hence the element $(4,2)$ in the signature. Finally, digit `4` has *four* predecessors (itself, `7`, `5` and `2`) and *four* successors (`7`, `5`, `2` and itself) hence the element $(4,4)$ in the signature.

The definition of the signature only considers predecessors and successors and can thus be naturally extended to traces. We give without a proof the following proposition:

Proposition 2. *The signature is invariant under the action of the group $\mathcal{S}_d$. Furthermore, two traces t_1 and t_2 have the same signature if and only if there exists a permutation $\sigma \in \mathcal{S}_d$ such that:*

$$t_2 = \sigma \cdot t_1$$

2.2 How Many PINs Are There?

We intend to count the number of PINs p such that $\tau(p) = t$. Let $\mathcal{P}$ be the pre-image of $\tilde{t}$ through the function $\tilde{\tau}$. Within each coset c_i in $\mathcal{P}$, there exists (at least) a PIN π_i such that $\tau(\pi_i) = t$.

Let p denote a PIN such that $t = \tau(p)$. We have $\tilde{t} = \tilde{\tau}(\tilde{p})$. This implies that there exists an index i such that $\tilde{p} = \tilde{\pi}_i$, which can be expressed as $p = \sigma \cdot \pi_i$.

Transposing to traces, we get:

$$t = \tau(p) = \tau(\sigma \cdot \pi_i) = \sigma \cdot \tau(\pi_i) = \sigma \cdot t.$$

Putting things together, the number of PINs p satisfying $\tau(p) = t$ is equal to the product of the number of cosets in the preimage of $\tilde{t}$ by the number of permutations σ such that $\sigma \cdot t = t$. We call the set of such permutations the *stabilizer* of t.

The signature of the trace t is an ordered set of couples of integers. This set can be permuted. The stabilizer of this signature consists in the permutations

leaving the set ordered. It is possible to prove that the stabilizer of the trace and the stabilizer of its signature have the same number of elements. The advantage here is that the signature's stabilizer is much easier to determine than the trace's stabilizer.

3 Evaluating the Entropy Loss Due to the Attack

To quantify the amount of secret information revealed by the attack, we denote by $w(p)$ the number of PINs q such that $\tau(p) = \tau(q)$. We need to evaluate for each integer n the function $e_\ell^d(n)$ counting the number of PINs satisfying $w(p) = n$. The observations made in the in the previous section allow to perform this task.

Step 1 Produce all the canonical PINs recursively. The function doing that is simply (Mathematica notation):

```
Rec[lst_, k_, n_] := Module[{i},
     If[k == 0, Treat[lst]; Return[]];
     For[i = 1, i ≤ n, i++, Rec[Append[lst, i], k - 1, n]];
     Rec[Append[lst, n + 1], k - 1, n + 1];
     ];
```

Note that whenever a canonical PIN is generated, `Rec` launches `Treat` on it.

Step 2 `Treat` Computes the signature of a canonical PIN. The intermediate variable `pre` contains the number of predecessors of each digit and `suc` contains the number of successors of each digit. Using `Transpose`, one obtains for each digit the number of its predecessors and successors. Sorting the so-obtained list yields the PIN's signature:

```
Treat[lst_] := Module[{t, l, s, i, j},
     l = Max[lst];
     t = 1;
     pre = Table[i, {i, 1, l}];
     For[i = 1, i ≤ Length[lst], i++,
        t = Max[t, lst[[i]]];
        pre[[lst[[i]]]] = t;
        ];
     suc = {};
     For[i = 1, i ≤ l, i++,
        s = 0;
        For[j = 1, j ≤ l, j++, If[pre[[j]] ≥ i, s++]];
        AppendTo[suc, s];
        ];
     τ = Sort[Transpose[{pre, suc}]];
     AppendTo[types, τ];
     ];
```

Step 3 We can now count for each signature the number of corresponding canonical PINs and multiply the result by the cardinality of the signature's stabilizer (given by `AutoSym`):

```
Nice[lst_] := Sort[({Length[Position[lst, #]], #}) & /@ Union[lst]];

AutoSym[lst_] := Times @@ ((#[[1]]!) & /@ Nice[lst]);

Compute[l_, d_] := Module[{ν, σ, nb, z, a, m, n},
     (* computing canonical PINs *)
     types = {};
     Rec[{1}, l - 1, 1];

     (* grouping traces *)
     ν = Nice[types];

     (* computing entropy *)
     nb = z = 0;
     For[i = 1, i ≤ Length[ν], i++,
        σ = AutoSym[ν[[i, 2]]];
        a = ν[[i, 1]]*σ;
        m = Max @@ ν[[i, 2]];
        n = ν[[i, 1]]*(d!/(d - m)!);
        nb += n;
        z += Log[2, a]*n;
        ];
     Print[N[z/nb, 20], " bits"];
     ];
```

Which evaluation (*e.g.* `In[1]:= Compute[9,10]`) yields:

```
5.2080553744037319192 bits
```

4 Results For Decimal PINs ($d = 10$)

In this section, we report for $3 \leq \ell \leq 8$, the number of PINs having a given w value and $H(\mathcal{P}_\ell^d)$, the amount of information (PIN entropy) not recovered by the chemical attack. The authors actually computed $e_\ell^{10}(n)$ for $3 \leq \ell \leq 12$ and all n values but the tables for $\ell \geq 9$ are too voluminous to be included here (68, 122, 226 and 429 nonzero n values were respectively found for ℓ = 9,10,11 and 12).

n	$\ell=3$	$\ell=4$	$\ell=5$	$\ell=6$	$\ell=7$	$\ell=8$
1	730	5770	45370	337690	2268010	13487050
2	270	1440	15120	120960	967680	7862400
3		2430				
4			20520	35280	635040	6713280
5				151650		
6			4320		907740	
7			5040			4234230
8		360		80640		
9				45360	816480	
10			7200	57600	655200	6048000
11					332640	5654880
12			1440	110880	181440	5564160
13						1965600
14					846720	
15					464400	
16						6652800
20					100800	
21						3190320
22			990			665280
24						483840
28						423360
30				21600		
32				23040	40320	1935360
35						1234800
36						1360800
38					383040	
44					332640	
47						4263840
48					483840	
52				2340		3144960
56					141120	1693440
58						7308000
60						1814400
68						1028160
70						2116800
84					60480	
102					73440	
108				12960		
114					5130	
120					201600	
128						967680
132						1995840
140						1411200
144					30240	362880
152						191520
198						1140480
240						10800
303						218160
336						1693440
456						1149120
600					72000	
720						1209600
2304						483840
2664						319680

Table 2. Values of $e_\ell^{10}(n)$.

ℓ	3	4	5	6	7	8	9	10	11	12
$H(\mathcal{P}_\ell^{10})$	0.27	0.63	1.15	1.84	2.74	3.86	5.21	6.80	8.62	10.68

Table 3. Values of $H(\mathcal{P}_\ell^{10})$.

4.1 Attacking Ratified PINs

PIN codes are usually protected against guessing by *ratification counters*. A ratification counter simply counts the number of presentations of false PINs and blocks the system as soon as this number reaches a threshold r. The following table lists the attacker's success probability for $d = 10$ as a function of ℓ and r.

Typically, in the case of usual ATMs ($\ell = 4, r = 3$), the attack will succeed in 98% of the cases. In GSM cards (where $\ell = 8, r = 3$) the attacker's success odds will still be 37%.

$\ell \rightarrowtail$	3	4	5	6	7	8	9	10	11	12
$r = 1$	0.865	0.734	0.604	0.469	0.339	0.226	0.137	0.074	0.035	0.014
$r = 2$	1.000	0.892	0.754	0.600	0.452	0.318	0.204	0.117	0.059	0.025
$r = 3$	1.000	0.978	0.829	0.671	0.517	0.370	0.242	0.142	0.073	0.032
$r = 4$	1.000	0.982	0.903	0.742	0.581	0.422	0.280	0.167	0.088	0.040
$r = 5$	1.000	0.986	0.926	0.804	0.629	0.458	0.305	0.184	0.098	0.045
$r = 6$	1.000	0.991	0.950	0.836	0.678	0.493	0.330	0.201	0.108	0.050
$r = 7$	1.000	0.996	0.966	0.868	0.711	0.529	0.355	0.217	0.118	0.055
$r = 8$	1.000	1.000	0.974	0.899	0.744	0.558	0.380	0.234	0.127	0.060

Table 4. Ratification Counter Probabilities for $d = 10$.

5 Countermeasures

A tactile screen keyboard where digits are assigned random positions seems to be the most efficient protection against the attack described in this paper. Low-tech but nonetheless efficient countermeasures consist in assigning a different finger to each key or, alternatively, keying the obfuscation sequence `0123456789876543210` before using the terminal...

References

1. `http://chipo.chem.uic.edu/web1/ocol/spec/MS1.htm`

APPENDIX
Canonical PINs for $d = 10$ and $\ell = 4$

canonical																			
0000	1111	2222	3333	4444	5555	6666	7777	8888	9999										
0001	0002	0003	0004	0005	0006	0007	0008	0009	1110	1112	1113	...	9998
0010	0020	0030	0040	0050	0060	0070	0080	0090	1101	1121	1131	...	9989
0011	0022	0033	0044	0055	0066	0077	0088	0099	1100	1122	1133	...	9988
0012	0013	0014	0015	0016	0017	0018	0019	0021	0023	0024	0025	...	9987
0100	0200	0300	0400	0500	0600	0700	0800	0900	1011	1211	1311	...	9899
0101	0202	0303	0404	0505	0606	0707	0808	0909	1010	1212	1313	...	9898
0102	0103	0104	0105	0106	0107	0108	0109	0201	0203	0204	0205	...	9897
0110	0220	0330	0440	0550	0660	0770	0880	0990	1001	1221	1331	...	9889
0111	0222	0333	0444	0555	0666	0777	0888	0999	1000	1222	1333	...	9888
0112	0113	0114	0115	0116	0117	0118	0119	0221	0223	0224	0225	...	9887
0120	0130	0140	0150	0160	0170	0180	0190	0210	0230	0240	0250	...	9879
0121	0131	0141	0151	0161	0171	0181	0191	0212	0232	0242	0252	...	9878
0122	0133	0144	0155	0166	0177	0188	0199	0211	0233	0244	0255	...	9877
0123	0124	0125	0126	0127	0128	0129	0132	0134	0135	0136	0137	...	9876