A Note on Key Rank

Daniel P. Martin[1] and Marco Martinoli[2]

[1] School of Mathematics, University of Bristol, Bristol, BS8 1TW, UK, and the Heilbronn Institute for Mathematical Research, Bristol, UK.*** dan.martin@bristol.ac.uk

[2] Department of Computer Science, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol, UK. marco.martinoli@bristol.ac.uk†

Abstract. In recent years key rank has become an important aspect of side-channel analysis, enabling an evaluation lab to analyse the security of a device after a side-channel attack. In particular, it enables the lab to do so when the enumeration effort would be beyond their computing power. Due to its importance there has been a host of work investigating key rank over the last few years.
In this work we build upon the existing literature to make progress on understanding various properties of key rank. We begin by showing when two different "scoring methods" will provide the same rank. This has been implicitly used by various algorithms in the past but here it is shown for a large class of functions. We conclude by giving the computational complexity of key rank. This implies that it is unlikely for, considerably, better algorithms to exist.

Keywords: Side-channel Attacks, Key Rank, Scoring Methods

1 Introduction

In 2013 Veyrat-Charvillion et al. [10] give the first key rank algorithm, which enabled evaluation labs to estimate the remaining security of a device, without having to enumerate all of the keys. In the half decade that has followed there has been a host of work producing alternate key rank algorithms providing different trade-offs between efficiency and accuracy [10,1,6,5,9,2,7].

While the majority of work uses probabilities, other works assume scores of a specific form arise from the side-channel distinguisher [9,2]. It has been stated in the literature that this makes research utilising scores not comparable with those which utilise probabilities [4]. In this work we show that for the majority scores required this is simply not the case.

The work of Martin et al. [9] converts the probabilities from the side-channel attack into additive integers, where the smallest is the most likely. They argue

*** This research was carried out while D. P. Martin was a member of the Department of Computer Science, University of Bristol.

† Funded by European Union's Horizon 2020 research and innovation programme Marie Sklodowska-Curie ITN ECRYPT-NET (Project Reference 643161).

why this is possible. This mechanism is improved in a follow up work [8]. Recently David and Wool [4] show that if a function f is applied to the probabilities, such that the probabilities can only increase, then the rank applied to these new scores will upperbound the actual rank. Similar techniques were recently used [7] to show the equivalence of the rank by Martin et al. [9] and that of Glowacz et al. [6].

Every time an equivalence property between different types of scores has been required, it has been shown specifically for the required example. This means that some of these ideas have been repeated several times, across various papers. In this paper we give a wide range of functions which can be applied to the scores whilst still preserving the rank. We also give a function that allows multiplicative scores to be converted to additive scores in a rank preserving manner. We conclude by giving the computational complexity of key rank. This implies we do not expect a more efficient key rank algorithm than the ones already in the literature, without a significant breakthrough in computational complexity.

1.1 Outline

In Section 2 we give any required notation and recap the definition of key rank. Section 3 gives a broad range of functions which can be applied to the output probabilities of a side-channel distinguisher while still keeping the rank invariant. In Section 4 we show the hardness of the key rank problem, showing that it is unlikely for a more efficient algorithm to exist.

2 Preliminaries

We recall notions, definitions and facts from the literature which are useful to contextualise our problem. These include basic notations from probability theory as well as an informal discussion on what is generally understood to be the key rank in the literature. We will conclude with its formal definition.

2.1 Random Variables

We denote vectors by lowercase boldface letters and we index them using subscript notation. In this paper we make exclusive use of discrete random variables defined over finite spaces. We denote them by uppercase italic letters, and write $X \sim \mathcal{D}$ to say a random variable X follows the distribution $\mathcal{D}$. We sometimes call the latter prior distribution when we want to highlight the difference with the distribution obtained by conditioning on some extra information $\mathcal{T}$. We denote the latter by $\mathcal{D}^{\mathcal{T}}$. The probability of a random variable X assuming a certain value x is defined as and denoted by

$$\Pr[X = x] = \mathcal{D}(x)$$

or just by $\Pr[x]$ if X is clear from the context. If additional information $\mathcal{T}$ is available, then we write

$$\Pr[X = x \mid \mathcal{T}] = \mathcal{D}^{\mathcal{T}}(x)$$

to express the probability conditioned on such information.

2.2 Key Rank

Every cryptographic primitive or protocol, once implemented, is vulnerable to implementation-specific attacks. These range from simple bugs exposing the secret in plaintext, to sophisticated attacks which exploit side-channels to infer something on the key being used. The latter scenario is the background and starting point of our work: we assume a cryptographically relevant operation, i.e. using a secret an adversary wishes to learn, is implemented and exposed to side-channels. The latter can be generically thought of as carriers of information on the secret key and we denote them by $\mathcal{T}$. Every time we use the variable $\mathcal{T}$, then, we implicitly assume the presence of some sort of dependency on the secret key.

Let $\mathcal{K}$ be the space of all possible subkeys, m be the number of subkeys (thus implying that the space of keys is $\mathcal{K}^m$) and $n = |\mathcal{K}|$ be the number of candidates per subkey[3]. A side-channel attack against the implementation of a cipher having $\mathcal{K}^m$ as key space is mounted and soft information are obtained. These can be described as being m posterior probability distributions conditioned on the leakage in the side-channels $\mathcal{T}$, one per subkey. We denote the random variable modeling the ith position of a key $\boldsymbol{k} \in \mathcal{K}^m$ by K_i, and by $\mathcal{D}_i^{\mathcal{T}}$ its posterior probability distribution. Such distribution can be thought of as being the ith marginal of a distribution $\mathcal{D}^{\mathcal{T}}$ over the entire key space.

Informally speaking, $\mathcal{D}_i^{\mathcal{T}}$ might tell if there are values which are very unlikely or very likely to be correct based on the side-channel information $\mathcal{T}$. The ideal case from an adversarial perspective is when each $\mathcal{D}_i^{\mathcal{T}}$ assigns the value 1 to the correct value in $\mathcal{K}$ and 0 to the others, meaning that the full secret key has been completely disclosed. We are now in a position to recap the definition of key rank.

Definition 1 (Key Rank). *The key rank of a key $\boldsymbol{k} \in \mathcal{K}^m$ is defined as the number of keys with a probability greater than $\boldsymbol{k}$. Formally*

$$\begin{aligned}\operatorname{rank}_{\mathcal{D}^{\mathcal{T}}}(\boldsymbol{k}) &= |\{\boldsymbol{t} \in \mathcal{K}^m : \Pr[\boldsymbol{k} \mid \mathcal{T}] < \Pr[\boldsymbol{t} \mid \mathcal{T}]\}| \\ &= \left|\left\{\boldsymbol{t} \in \mathcal{K}^m : \Pr\left[\bigwedge_{i=1}^{m}(K_i = k_i) \,\middle|\, \mathcal{T}\right] < \Pr\left[\bigwedge_{i=1}^{m}(K_i = t_i) \,\middle|\, \mathcal{T}\right]\right\}\right| .\end{aligned}$$

[3] Our work does not require all subkeys to be the same size. However, we do so to ease explanation.

In this work, as with the majority of work in the key rank literature, we assume that the leakage on each of the subkeys is independent, i.e. that

$$\Pr\left[\bigwedge_{i=1}^{m}(K_i = k_i) \;\middle|\; \mathcal{T}\right] = \prod_{i=1}^{m}\Pr[(K_i = k_i) \mid \mathcal{T}] \ .$$

While our definition does not need such assumption, it will be required in the analysis that follows.

Definition 1 is very formal in its use of posterior probabilities on single subkeys to derive a rank for the full key. Often, practical reasons forbid a designer to adopt such a strict definition, and is easier and more useful to use an approximation of probabilities called *scores*. We denote the score the ith subkey k_i assumes on value $j \in \mathcal{K}$ by $S_{i,j}$ and the function returning the score of a full key by $S(\boldsymbol{k})$. They can be quite different from probabilities, in particular they can be additive rather than multiplicative. We therefore give an alternative definition of key rank for additive scores.

Definition 2 (Additive Key Rank). *The additive key rank of a key $\boldsymbol{k} \in \mathcal{K}^m$ is defined as the number of keys with a score greater than $\boldsymbol{k}$. Formally*

$$\begin{aligned}\mathrm{rank}^{+}_{\mathcal{D}^{\mathcal{T}}}(\boldsymbol{k}) &= |\{\boldsymbol{t} \in \mathcal{K}^m : S(\boldsymbol{k}) < S(\boldsymbol{t})\}| \\ &= \left|\left\{\boldsymbol{t} \in \mathcal{K}^m : \sum_{i=1}^{m} S_{i,k_i} < \sum_{i=1}^{m} S_{i,t_i}\right\}\right| \ .\end{aligned}$$

For the remaining of this paper, we avoid conditioning on $\mathcal{T}$ whenever we write a probability, thus leaving it implicit. We do this for the sake of notational convenience.

3 Equivalence

Our first contribution lies in formally defining under which circumstances the rank of a key remains unmodified even if the representation used to compute it does change. In other words, we define a class of functions that modify the underlying probabilities or scores without affecting the relative order of candidate keys, thus preserving the notion of rank.

Definition 3 (Rank Preserving). *Let f be a function over the reals. Then f is called rank preserving if, for all $\boldsymbol{k} \in \mathcal{K}^m$*

$$\mathrm{rank}_{f(\mathcal{D}^{\mathcal{T}})}(\boldsymbol{k}) = \mathrm{rank}_{\mathcal{D}^{\mathcal{T}}}(\boldsymbol{k})$$

where $f(\mathcal{D}^{\mathcal{T}})$ represents applying the function f to every $\mathcal{D}^{\mathcal{T}}_i(j)$. Equivalently, f is additive rank preserving if, for all $\boldsymbol{k} \in \mathcal{K}^m$

$$\mathrm{rank}^{+}_{f(\mathcal{D}^{\mathcal{T}})}(\boldsymbol{k}) = \mathrm{rank}^{+}_{\mathcal{D}^{\mathcal{T}}}(\boldsymbol{k}) \ .$$

In this section we show the class of functions which can be applied to the probabilities, without adjusting the rank of the key. We stress that this makes probabilities and their function thereof comparable.

Theorem 1. *Functions in the class $f(x) = a{\cdot}x^b$ for $a, b \in \mathbb{R}$ are rank preserving.*

Proof. For a key $\boldsymbol{t}$ to be in the set counted by $\text{rank}_{f(\mathcal{D}^\mathcal{T})}(\boldsymbol{k})$, we have that:

$$\begin{aligned}
\prod_{i=1}^{m} f(\Pr[k_i]) &< \prod_{i=1}^{m} f(\Pr[t_i]) \\
\prod_{i=1}^{m} a \cdot \Pr[k_i]^b &< \prod_{i=1}^{m} a \cdot \Pr[t_i]^b \\
a^m \prod_{i=1}^{m} \Pr[k_i]^b &< a^m \prod_{i=1}^{m} \Pr[t_i]^b \\
\prod_{i=1}^{m} \Pr[k_i]^b &< \prod_{i=1}^{m} \Pr[t_i]^b \\
\left(\prod_{i=1}^{m} \Pr[k_i]\right)^b &< \left(\prod_{i=1}^{m} \Pr[t_i]\right)^b \\
\prod_{i=1}^{m} \Pr[k_i] &< \prod_{i=1}^{m} \Pr[t_i] \ .
\end{aligned}$$

Thus the key $\boldsymbol{t}$ is also counted by $\text{rank}_{\mathcal{D}^\mathcal{T}}(\boldsymbol{k})$. □

A regularity result holds when scores are defined as the logarithm of probabilities. In particular, a connection between rank and additive rank exists, as proven in the following theorem.

Theorem 2. *Given $f(x) = \log x$, for all $\boldsymbol{k} \in \mathcal{K}^m$ we have that*

$$\text{rank}_{\mathcal{D}^\mathcal{T}}(\boldsymbol{k}) = \text{rank}^{+}_{f(\mathcal{D}^\mathcal{T})}(\boldsymbol{k}) \ .$$

Proof. For a key $\boldsymbol{t}$ to be in the set counted by $\text{rank}^{+}_{f(\mathcal{D}^{\mathcal{T}})}(\boldsymbol{k})$, we have that:

$$\begin{aligned}
\sum_{i=1}^{m} S_{i,k_i} &< \sum_{i=1}^{m} S_{i,t_i} \\
\sum_{i=1}^{m} f(\Pr[k_i]) &< \sum_{i=1}^{m} f(\Pr[t_i]) \\
\sum_{i=1}^{m} \log \Pr[k_i] &< \sum_{i=1}^{m} \log \Pr[t_i] \\
2^{\sum_{i=1}^{m} \log \Pr[k_i]} &< 2^{\sum_{i=1}^{m} \log \Pr[t_i]} \\
\prod_{i=1}^{m} 2^{\log \Pr[k_i]} &< \prod_{i=1}^{m} 2^{\log \Pr[t_i]} \\
\prod_{i=1}^{m} \Pr[k_i] &< \prod_{i=1}^{m} \Pr[t_i] \ .
\end{aligned}$$

Thus the key $\boldsymbol{t}$ is also counted by $\mathsf{rank}_{\mathcal{D}}(\boldsymbol{k})$. □

The above two theorems can be combined to give the following corollary for additive rank.

Corollary 1. *Functions in the class $f(x) = a + bx$ for $a, b \in \mathbb{Z}^{+}$ are additive rank preserving.*

There are two logical ways in which key rank can be defined. The first is as we have done above where the rank only counts keys with a strictly better score. The second is where keys with an equal score is also counted. We use the former as it is the most commonly used in the literature and assumes a more powerful adversary. However, the latter additionally leads to the following nice result, which does not hold for the former.

Theorem 3. *Let f be a rank preserving function and let $\lceil x \rceil$ be the ceiling function, then:*

$$\begin{aligned}
\text{rank}_{f(\mathcal{D}^{\mathcal{T}})}(\boldsymbol{k}) &\leq \text{rank}_{\lceil f(\mathcal{D}^{\mathcal{T}}) \rceil}(\boldsymbol{k}) \\
\text{rank}^{+}_{f(\mathcal{D}^{\mathcal{T}})}(\boldsymbol{k}) &\leq \text{rank}^{+}_{\lceil f(\mathcal{D}^{\mathcal{T}}) \rceil}(\boldsymbol{k}) \ .
\end{aligned}$$

Proof. The first statement follows directly from Proposition 1 of David and Wool [4]. The second statement follows from the first and monotonicity of the logarithm function. □

Finally, we note that any function ℓ applied to the final product of probabilities, $\ell(\prod_{i=1}^{m} \Pr[k_i])$, trivially preserves the rank providing that the function is monotonically increasing.

3.1 Functions of Subkeys

All classes of functions we have provided a proof for were applied to full key ranks. In certain scenarios, it might be interesting to also investigate functions applied to probabilities of subkeys.

Definition 4 (Order Preserving). *Let f be a function over the reals and let us write $\mathcal{K} = \{j_1, \dots j_n\}$. Then f is called order preserving if for all $i \leq m$ it holds that*

$$f(\Pr[k_i = j_1]) < \ldots < f(\Pr[k_i = j_n]) \Leftrightarrow \Pr[k_i = j_1] < \ldots < \Pr[k_i = j_n] \ .$$

The above definition is particularly useful when the extend-and-prune [3] technique is applied, i.e. when positions are targeted sequentially and leakage on later ones include a subset of guesses for previous ones. The size of the subset is determined by a *pruning strategy*, which is deployed to discard (a.k.a. prune) particularly unlikely guesses.

After the ith position has been targeted and a posterior distribution has been obtained for that particular position (including information on all the previous ones), an attacker has to decide how many to prune based on a pruning strategy. To the best of our knowledge, there is no comprehensive study comparing several strategies, and a subset of them involve making a decision based on the order of the probabilities $\Pr[k_i = j]$. Thus, every function preserving the relative order of posterior probabilities does not affect the pruning strategies. Depending on the function f, strategies relying on the actual values of the probabilities, rather than just on their order, might be unaffected too.

We investigate this matter no further because we consider it out of scope for the current paper as preserving the order of probabilities for subkeys has no guarantee of preserving the rank: the latter crucially relies on values of probabilities of subkeys (or scores thereof), meaning that if these are changed despite the order being preserved, the final rank will likely be different.

Theorem 4. *There exists a function f which is order preserving but which is not rank preserving.*

Proof. When $m = 1$ the two definition perfectly coincide, hence we fix $m = n = 2$. There are then two subkeys k_1 and k_2 which can assume two possible values. We denote them by j_1 and j_2. We summarise in the following table the posterior probabilities once an attack has taken place.

	k_1	k_2
j_1	0.4	0.2
j_2	0.6	0.8

The correct key is $\boldsymbol{k} = (j_1, j_2)$, which has rank 2 since the key (j_2, j_2) has an higher product of probabilities. We now define $f(x) = x - 0.7$. It is trivially order

preserving because it simply applies a translational constant to each probability, hence leaving the order in each column unchanged. However, we now obtain that $\boldsymbol{k}$ has rank 4 because all other keys have an higher product of scores. □

4 Hardness of Key Rank

In this section we discuss the version of key rank where all the scores are positive integers which combine additively and the smaller the integer the more likely the subkey. From Section 3, this is equivalent to the original definition of key rank, utilising probabilities. We denote this as $\mathbb{Z}$-rank^+.

Definition 5 (#P). *The class of function problems of the form "compute f(x)", where f is the number of accepting paths of an NP machine.*

Lemma 1. *#Knapsack is #P-Complete.*

Theorem 5. *The #multiple-choice knapsack problem is #P-Complete.*

Proof. This can be shown by reducing the #knapsack problem to the #multiple-choice knapsack problem.

Given the set of items $\{x_i\}_{i=1}$, the reduction creates the set of sets $\{\{x_i, 0\}\}_{i=1}^{n}$. The algorithm to solve #multiple-choice knapsack problem is then run on this problem instance. Since exactly one item can be added per set, the solution to the #multiple-choice knapsack problem is trivially the solution to the #knapsack problem as follows. For the i^{th} set if x_i was chosen, x_i would chosen for the knapsack problem, while if 0 was chosen, this corresponds to not choosing x_i. Therefore every possible solution to the multiple-choice version corresponds to a solution to the original version. □

Theorem 6. $\mathbb{Z}$-rank^+ *is #P-Complete.*

Proof. This can be shown by reducing the #multiple-choice knapsack problem to $\mathbb{Z}$-rank^+.

The only difference between #multiple-choice knapsack problem and $\mathbb{Z}$-rank^+ is that the former takes in a weight directly, while the latter takes in a key which can be seen as an index into the sets to choose a weight. Therefore we must embed the desired weight W into the sets. To do this an extra set is added which contains $W - \sum_i min_j x_{i,j}$ and $m-1$ zeros (if each) set has m elements. The key is then the indices to to these minimum values plus the index to $W - \sum_i min_j x_{i,j}$.

This is then passed to the $\mathbb{Z}$-rank^+ algorithm. The result is calculate by diving the output by $m-1$. Any solution cannot contain $W - \sum_i min_j x_{i,j}$ because the total minimum weight would be at least W so wouldn't be counted. Any solution involving a 0 in the last set has an equivalent solution involving any of the other zeros in the final set. Thus the solution follows. □

References

1. Daniel J. Bernstein, Tanja Lange, and Christine van Vredendaal. Tighter, faster, simpler side-channel security evaluations beyond computing power. Cryptology ePrint Archive, Report 2015/221, 2015. `http://eprint.iacr.org/2015/221`.
2. Andrey Bogdanov, Ilya Kizhvatov, Kamran Manzoor, Elmar Tischhauser, and Marc Witteman. Fast and memory-efficient key recovery in side-channel attacks. Cryptology ePrint Archive, Report 2015/795, 2015. `http://eprint.iacr.org/2015/795`.
3. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES 2002*, volume 2523 of *LNCS*, pages 13–28. Springer, Heidelberg, August 2003.
4. Liron David and Avishai Wool. Prank: Fast analytical rank estimation via pareto distributions. Cryptology ePrint Archive, Report 2018/550, 2018. `https://eprint.iacr.org/2018/550`.
5. Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete - or how to evaluate the security of any leaking device. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 401–429. Springer, Heidelberg, April 2015.
6. Cezary Glowacz, Vincent Grosso, Romain Poussier, Joachim Schüth, and François-Xavier Standaert. Simpler and more efficient rank estimation for side-channel security assessment. In Gregor Leander, editor, *FSE 2015*, volume 9054 of *LNCS*, pages 117–129. Springer, Heidelberg, March 2015.
7. Daniel P Martin, Luke Mather, and Elisabeth Oswald. Two sides of the same coin: counting and enumerating keys post side-channel attacks revisited. In *Cryptographers Track at the RSA Conference*, pages 394–412. Springer, 2018.
8. Daniel P Martin, Luke Mather, Elisabeth Oswald, and Martijn Stam. Characterisation and estimation of the key rank distribution in the context of side channel evaluations. In *Asiacrypt 2016*, volume 10031, pages 548–572. Springer, 2016.
9. Daniel P. Martin, Jonathan F. O'Connell, Elisabeth Oswald, and Martijn Stam. Counting keys in parallel after a side channel attack. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 313–337. Springer, Heidelberg, November / December 2015.
10. Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Security evaluations beyond computing power. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 126–141. Springer, Heidelberg, May 2013.