Simon's Circuit

A Note on Cleverly-Chosen Circuits

Paul Baecher*

Abstract

Simon mentions in his seminal result separating collision-resistant hash functions from one-way permutations (EUROCRYPT '98), that the wrong strategy to sample collisions can be exploited to invert the permutation. He, however, does not spell out a concrete circuit that demonstrates this. In this short note, we describe and analyze one such circuit.

1 Introduction

Consider the following two collision-sampling oracles.

Oracle $\mathcal{O}_{\mathsf{A}}$ takes as input a circuit, samples uniformly at random a point in the domain of the circuit, and samples a second point (again uniformly) from the set of points that map to the same output. It returns the two points.

Oracle $\mathcal{O}_{\mathsf{B}}$ takes as input a circuit and samples uniformly at random from the list of all collisions. It outputs the collision.

Which one of the oracles can you exploit to invert a one-way permutation?

Simon answers this question in his oracle separation [Sim98], where collision-resistant hash functions do not exist, because a break oracle simply returns collisions for any circuit input. He writes [Sim98] on page 4:

> *For example, an oracle which simply returns a random collision (i.e., a uniformly chosen entry in the list of colliding pairs) would allow f to be inverted. (The space of collisions can be manipulated by use of a cleverly chosen query circuit, so that a constant fraction of all collisions involve an exponentially small fraction of the outputs.)*

Put differently, oracle $\mathcal{O}_{\mathsf{B}}$ is not a good choice. In the remainder of this note, we study such a "cleverly chosen" circuit—a circuit that allows us to invert a one-way permutation in the presence of oracle $\mathcal{O}_{\mathsf{B}}$. (But of course not in the presence of oracle $\mathcal{O}_{\mathsf{A}}$, since Simon shows that *no* circuit achieves this.)

Overview Say you would like to invert a one-way permutation π on some given point $y = \pi(x)$. Since $\mathcal{O}_{\mathsf{B}}$ simply samples from a large list of collisions of some circuit C, the main idea is to make sure that the preimage x under π shows up in the list of collisions so often, that the oracle returns it with good probability. At the same time, however, we know there cannot be many

*Darmstadt University of Technology. E-Mail: baecher@cs.tu-darmstadt.de.

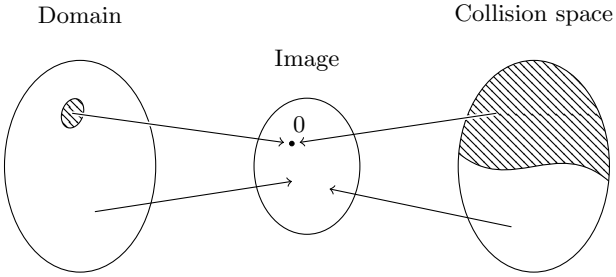

Figure 1: Even though only a very small preimage subset maps to the zero point, the collision space is evenly split between collisions that map to zero and those that do not. The hatched areas encode the permutation's preimage x of y.

preimages of C that contain x. Otherwise, oracle $\mathcal{O}_{\mathsf{A}}$ would be just as useful towards inverting π, but we know from [Sim98] that this cannot be the case.

A key insight that allows us to balance these two competing constraints is the following. For any function, the number of preimages $k := k(z)$ for a given image value z leads to roughly k^2 entries in the collision list related to z. This is because all combinations of two preimages mapping to the same value are feasible collisions. Depending on the exact definition of a collision, that results in $k(k-1)/2$ to k^2 pairs. It turns out that this gap, namely k preimages versus k^2 collisions, enables us to satisfy both contraints.

Our circuit is essentially the identity function (modulo compression) on $2n+1$ bits, except for one special image value 0^{2n}; see Figure 1. This value has 2^n preimages—a tiny fraction of the domain—and all of them encode the value x we would like to recover. Yet, the number of collisions induced by these preimages is $2^{n\,2} = 2^{2n}$, one half of the entire collision space. Hence, oracle $\mathcal{O}_{\mathsf{B}}$ returns a collision containing x with constant probability.

2 Collisions

For any function f, the tuple (x_0, x_1) is called a *collision* if $f(x_0) = f(x_1)$. Note that x_0 and x_1 need not be distinct, as it is the case in [Sim98]. Indeed, if $x_0 = x_1$, we call the collision *trivial*. Note furthermore that two collisions (x_0, x_1) and (x_1, x_0) are considered different if $x_0 \neq x_1$.

3 The Circuit

Denote by $\pi : \{0,1\}^n \to \{0,1\}^n$ a one-way permutation and let $y \in \{0,1\}^n$. Define the oracle circuit $C_y^\pi : \{0,1\} \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^n$ as

$$C_y^\pi(b, x, x') := \begin{cases} (0^n, 0^n) & \text{if } \pi(x) = y, \\ (x, x') & \text{otherwise.} \end{cases}$$

Clearly, the circuit is compressing.

Let us formally analyze the collision space of the circuit with respect to our definition of a collision. We begin with the second case of the circuit. This case induces collisions, including trivial ones, of the form

$$((b_0, x, x'), (b_1, x, x'))$$

as (b_0, b_1, x, x') ranges over

$$\{0,1\} \times \{0,1\} \times \left(\{0,1\}^n \setminus \{\pi^{-1}(y)\}\right) \times \{0,1\}^n.$$

We have $2 \cdot 2 \cdot (2^n - 1) \cdot 2^n = 2^{2n+2} - 2^{n+2} \in O(2^{2n})$ such collisions which we denote by the set Coll_2.

The first case of the circuit induces collisions, again including trivial ones, of the form

$$\left((b_0, \pi^{-1}(y), x_0), (b_1, \pi^{-1}(y), x_1)\right)$$

where

$$(b_0, x_0, b_1, x_1) \in \{0,1\} \times \{0,1\}^n \times \{0,1\} \times \{0,1\}^n.$$

There are $2^{2n+2} \in \Theta(2^{2n})$ collisions of this form; denote these collisions as Coll_1.

Finally, there are some collisions that involve both cases. These collisions are of the forms

$$\left((b_0, 0^n, 0^n), (b_1, \pi^{-1}(y), x)\right), \quad \text{or} \quad \left((b_1, \pi^{-1}(y), x), (b_0, 0^n, 0^n)\right),$$

where

$$(b_0, b_1, x) \in \{0,1\} \times \{0,1\} \times \{0,1\}^n.$$

This accounts for $2 \cdot (2 \cdot 2 \cdot 2^n) = 2^{n+3} \in \Theta(2^n)$ such collisions in the set Coll_{12}. Note that, by construction, all sets of collisions are disjoint.

4 Two Collision-Sampling Oracles

Let us now analyze the usefulness of two different collision-sampling strategies $\mathcal{O}_\mathsf{A}$, $\mathcal{O}_\mathsf{B}$ for inverting π. Note that the analysis for $\mathcal{O}_\mathsf{A}$ is technically redundant (yet instructive), because it is a special case of Simon's main result.

Oracle $\mathcal{O}_\mathsf{A}(C_y^\pi)$:	Oracle $\mathcal{O}_\mathsf{B}(C_y^\pi)$:
1 $(b_0, x_0, x_0') \leftarrow_\$ \{0,1\} \times \{0,1\}^n \times \{0,1\}^n$	1 $X \leftarrow \{(v, v') : C_y^\pi(v) = C_y^\pi(v')\}$
2 $z \leftarrow C_y^\pi(b_0, x_0, x_0')$	2 $c \leftarrow_\$ X$
3 $X \leftarrow \{(b, x, x') : C_y^\pi(b, x, x') = z\}$	3 return c
4 $(b_1, x_1, x_1') \leftarrow_\$ X$	
5 return $((b_0, x_0, x_0'), (b_1, x_1, x_1'))$	

Oracle $\mathcal{O}_\mathsf{A}$. This oracle first samples a point uniformly in the domain of the circuit and then samples uniformly a second, colliding preimage. For this sampling process, define event G_0 as "hitting" the first case in the first line, i.e., $\pi(x_0) = y$. Analogously, define G_1 as hitting the first case in the fourth line, i.e., $\pi(x_1) = y$. We also define event Z to occur if $(x_0, x_0') = (0^n, 0^n)$. Clearly, the oracle's answer can only be used to invert the one-way permutation if either G_0 or G_1 happens: otherwise, the outputs are independent of π. We define this by the event Invert and observe that

$$\begin{aligned}
\Pr[\mathsf{Invert}] &= \Pr[\mathsf{G}_0 \vee \mathsf{G}_1] \\
&= \Pr[\mathsf{G}_0] + \Pr[\mathsf{G}_1 \wedge \neg\mathsf{G}_0] \\
&= \Pr[\mathsf{G}_0] + \Pr[\mathsf{G}_1 \wedge \neg\mathsf{G}_0 | \mathsf{Z}] \Pr[\mathsf{Z}] + \Pr[\mathsf{G}_1 | \neg\mathsf{G}_0 \wedge \neg\mathsf{Z}] \Pr[\neg\mathsf{G}_0 \wedge \neg\mathsf{Z}] \\
&= \Pr[\mathsf{G}_0] + \Pr[\mathsf{G}_1 \wedge \neg\mathsf{G}_0 | \mathsf{Z}] \Pr[\mathsf{Z}] \\
&\leq \Pr[\mathsf{G}_0] + \Pr[\mathsf{Z}] \\
&= 2^{-n} + 2^{-2n}.
\end{aligned}$$

Here we used that $\Pr[\mathsf{G}_1|\neg\mathsf{G}_0 \wedge \neg\mathsf{Z}] = 0$. To see this, note that $\neg\mathsf{G}_0$ implies $x_0 \neq \pi^{-1}(y)$. Thus, it follows that $C_y^\pi(b_0, x_0, x_0') = (x_0, x_0')$ in the second line, and, since $\neg\mathsf{Z}$, we know that $(x_0, x_0') \neq (0^n, 0^n)$. The set of possible colliding points X in the third line hence only contains $(0, x_0, x_0')$ and $(1, x_0, x_0')$; therefore $x_1 = x_0 \neq \pi^{-1}(y)$, so G_1 cannot happen. We conclude that $\mathcal{O}_\mathsf{A}$ will return a preimage with negligible probability only.

Oracle $\mathcal{O}_\mathsf{B}$. This oracle samples uniformly from the list of all collisions. For the circuit defined in the previous section this means that, with probability roughly $\frac{1}{2}$, the oracle will output a collision from the set Coll_1 (or Coll_2) because

$$2^n \approx |\mathsf{Coll}_{12}| \ll |\mathsf{Coll}_1| \approx |\mathsf{Coll}_2| \approx 2^{2n}.$$

However, each collision from Coll_1 contains the value $\pi^{-1}(y)$, i.e., the unique preimage of y. Therefore, the oracle can be used to invert π with constant probability.

Remark. The curious reader should convince herself that the same arguments can be made go through if (a) one does not count trivial collisions (x, x) as proper collisions, and (b) if one considers collisions as sets $\{x_0, x_1\}$ instead of pairs (x_0, x_1). (Or a combination thereof.)

Acknowledgements

I thank Christina Brzuska, Pooya Farshim, Giorgia Azzurra Marson, and Arno Mittelbach for valuable comments on earlier versions of this article.

References

[Sim98] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 334–345. Springer, May / June 1998.