Space-Time Tradeoff in Finding Collisions in Cryptographic Hash Functions

Maria Polyakova

June 6, 2013

Background

The space-time tradeoff is a situation where memory use and time are inversely proportional to each other. A problem may complete faster when more memory is available, and with less memory, time increases.

To demonstrate this concept, an analysis is performed on finding collisions on the first 32 bits of the MD5 hash function, however, can be repeated on any hash function. The analysis is done to a probability of .5: at that probability, in half of the cases the collision would have been found. This is an arbitrary constant, and the tests can be performed with any probability less than 1.

The simulation happens in two stages. Stage one is the filling up of the memory table, and calculating the probability of finding a collision there. When it is full, stage two happens: continuously trying values until a collision is found(with .5 percent probability). An assumption is made that the input is not random: once an input is tried it is not entered again, however, that different inputs can hash to the same value. The full output, therefore, the distribution is geometric: the average number of attempts that must be made before a collision is found is the expected value of the geometric distribution: $1/p$ where p is the probability of success-probability that a random chosen value yields a collision.

Stage 1 Description

At stage one, the memory table is being filled up. Every table entry must hold two values: the first 32 bits of the hash, and the message that created it. Currently, however, we will only be looking at the number of entries we can store.

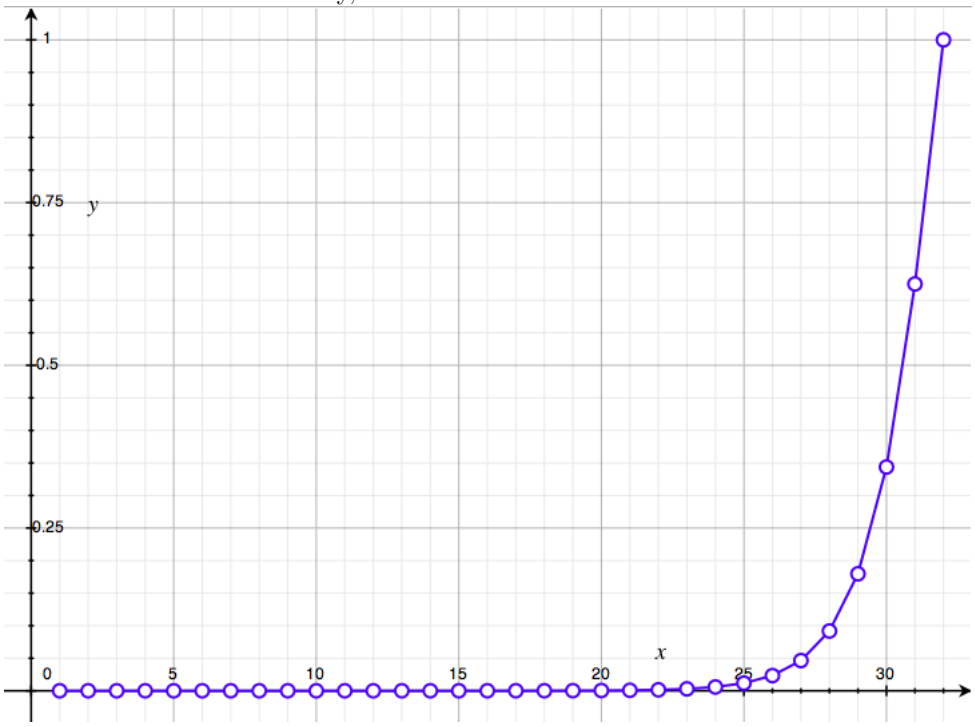
Every time a value is added to the memory table, we must see if that value is already present in the table: if a collision has already occurred, before appending it. Every time a value is appended to the table, the probability of randomly successfully finding a collision grows:

$$P = \frac{x}{N}$$

where P is the probability of finding a collision with 1 guess, x is the number of entries in the table, and N is size of the total space of the output, in our case 2^{32}.
This being said, every time a value is added to the table, the probability of the next value giving a collision is increased.
This can be modeled by the following function, negated for easier computation:

$$1 - \prod_{i=1}^{x} \frac{N-i}{N}$$

where x is size of the memory, and N is as defined before.



x-axis is the size of the memory log base 2, also the number of attempts. number of slots in the memory is 2^x. Y scale is the probability of getting a collision at that point.

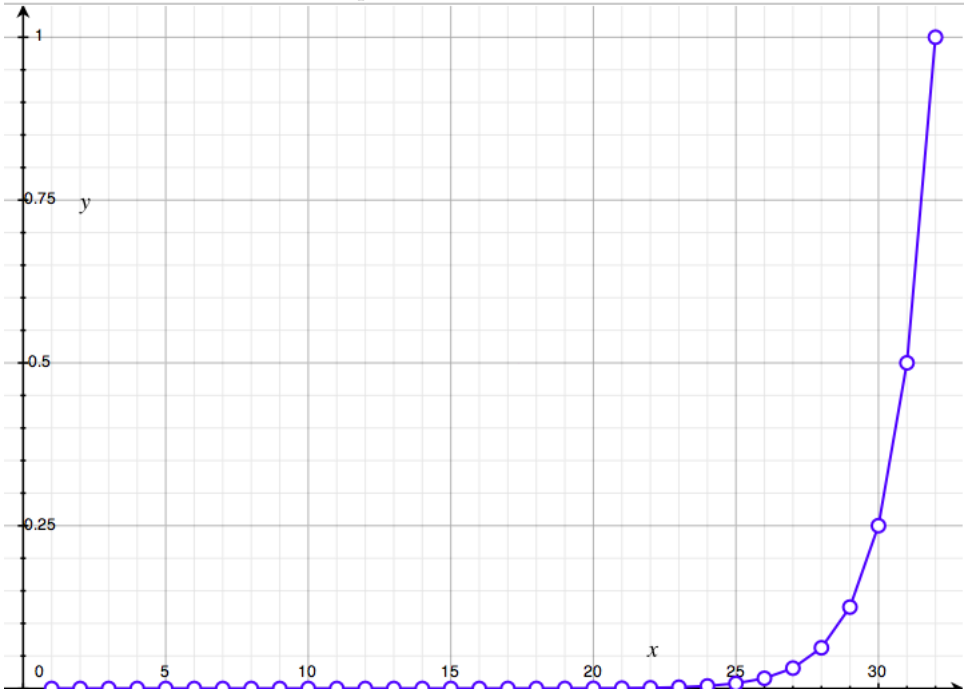
Stage 2 Description

At stage two, the memory is already full, and from now on, values are compared to values in the table. Since messages can hash to any value, upon trying a value, the possible set doesn't get any smaller, all 2^32 values are still possible.

At every attempt, the probability of finding a collision is:

$$\frac{X}{N}$$

where X is the size of the memory(as before, the number of slots how filled in), and N is the size of the total space: 2^{32}.



graph of probabilities of success vs size in bits

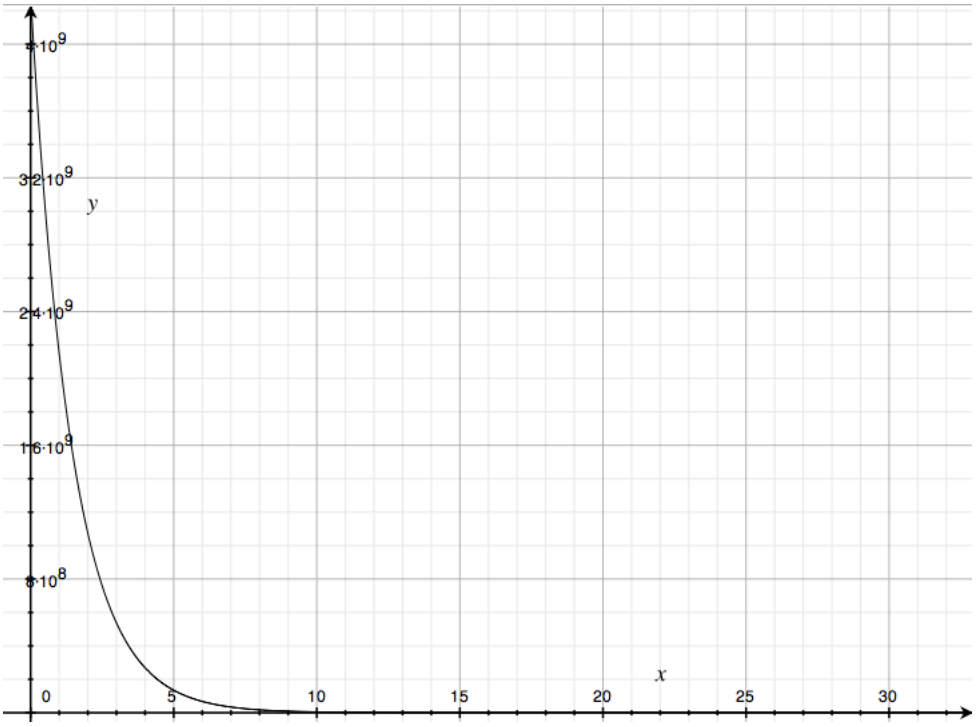
Each trial is a Bernoulli trial: a trial with two outcomes, success and failure. The distribution is geometric, with the following probability of the first success on kth trial:

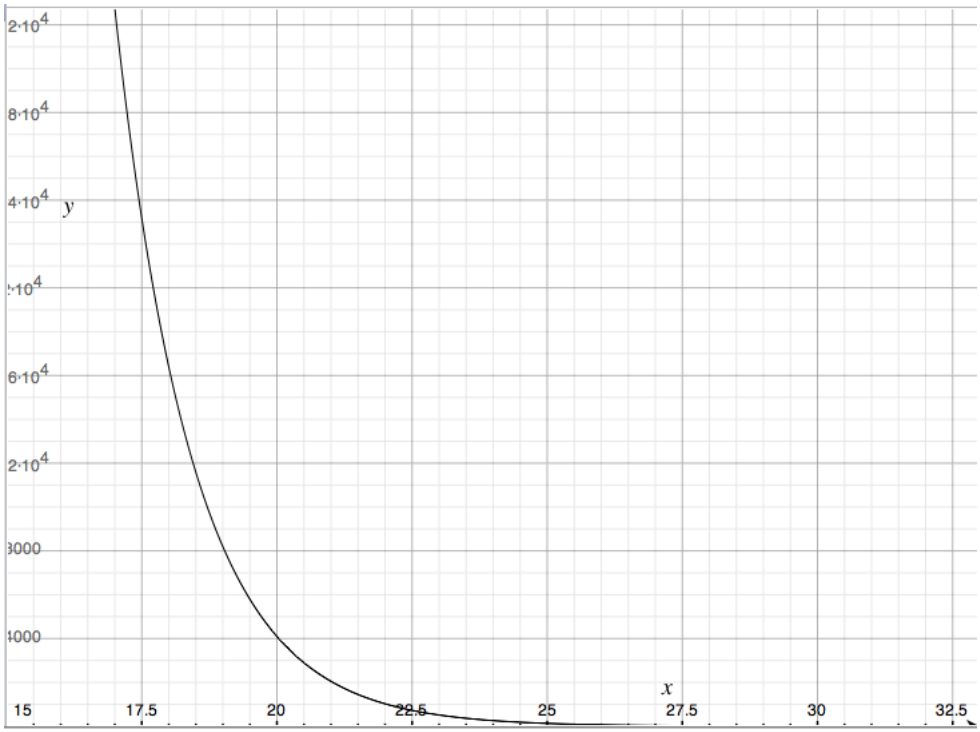
$$Pr(X = k) = (1 - p)^{k-1}p$$

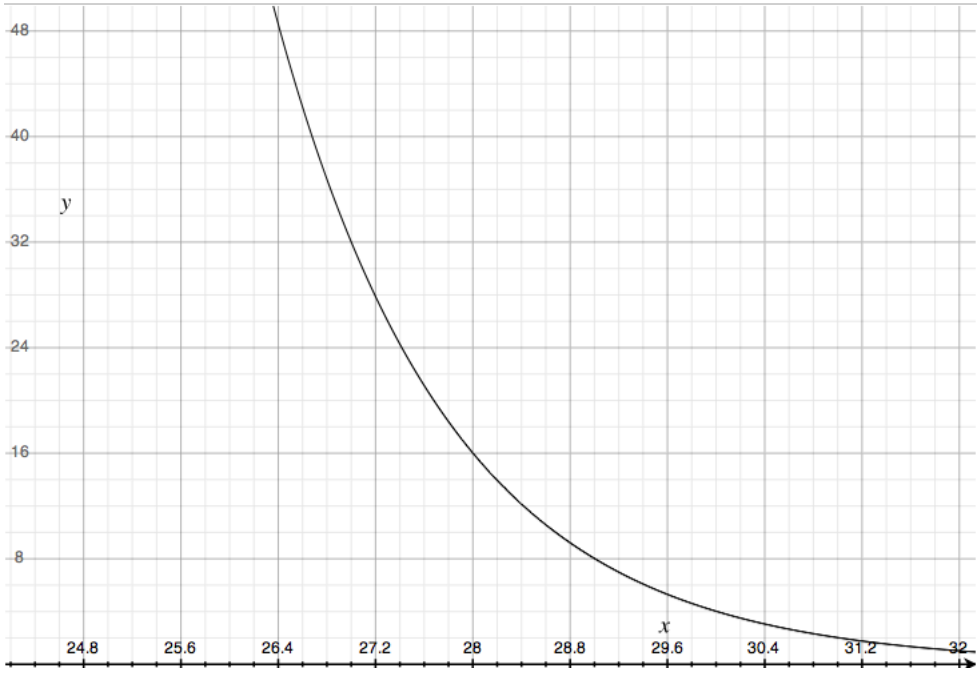
p is the probability of success, as described above. This value stays constant, as the table is already full.
The probability is .5, the variable we're missing is k: the number of attempts that occur before finding a collision(with .5 probability).
For finding the number of attempts, the expected value can be used. Because the output of a cryptographic hash function is evenly distributed, the expected value will show the very middle of the distribution: .5.
The expected value of a geometric distribution(as above) is given by:

$$E[X] = \frac{1}{p}$$

where x is the number of trials, the k, and p is the probability of success.
In the graphs below, the expected number of attempts necessary after the table has been completely filled in with no collisions.
x axis is 2^x filled slots of memory. y axis is the number of attempts needed on average to find a collision







same axes as above

In the graph above, it is visible that at memory of 2^{32} entries, the very next attempt will find a collision. This is the birthday problem

Putting Stages Together

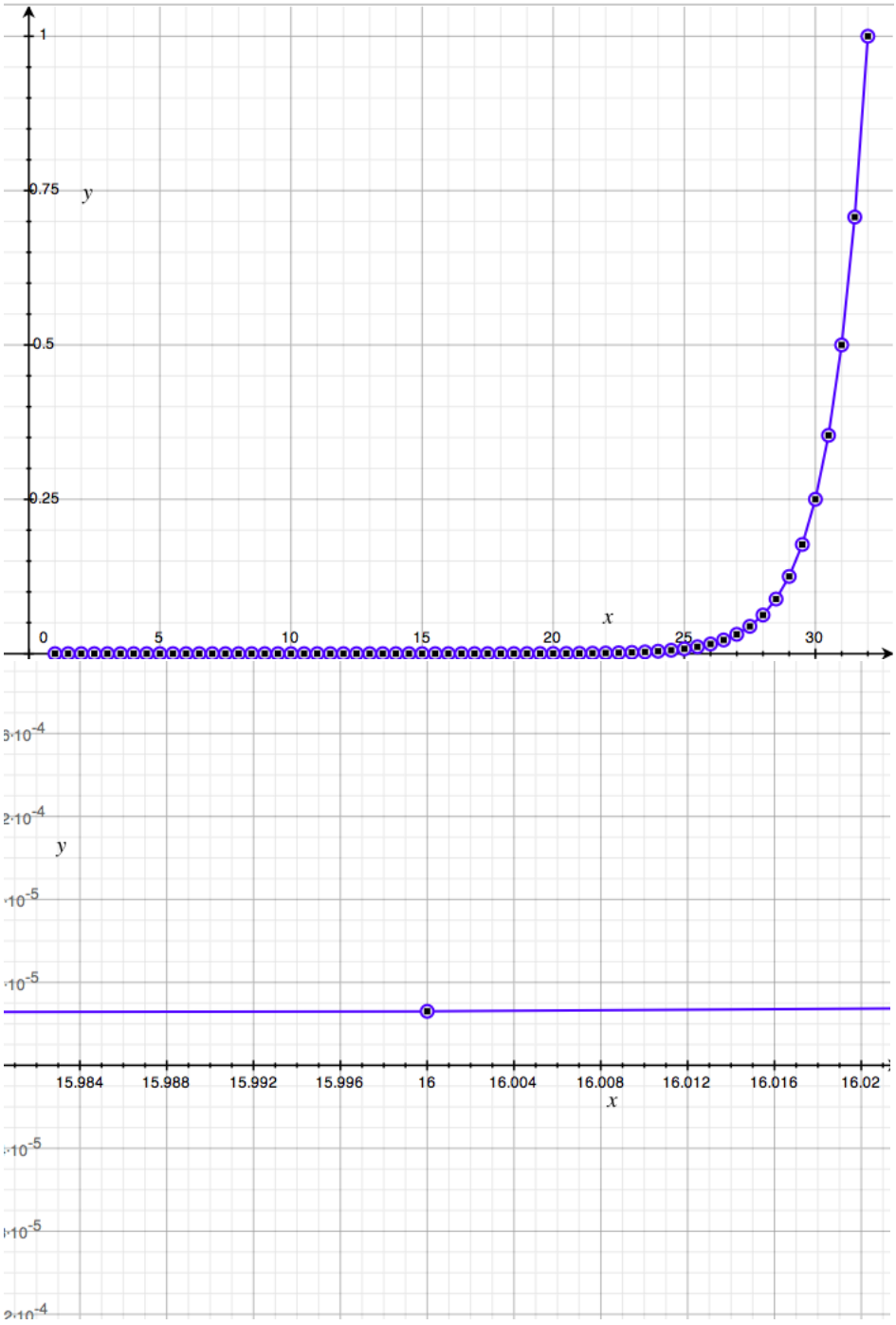
Stage 1 ends when all memory slots are filled in. Until then it follows the equation given in Stage 1. After that, it follows the equation given in Stage 2.
Probability of x attempts:
T = slots available for storage
$N = 2^{32}$

$$Pr(X = k) = \begin{cases} 1 - \prod_{i=1}^{x} \frac{N-i}{N} & : x < T \\ \\ 1 - (1-p)^{x-T} & : x > T \end{cases}$$

x is the number of attempts, translating directly to time. The average number of attempts it would take to find a collision is the x at which p=.5.
(Or any other value chosen).
If the number of available slots is smaller than 2^{32}, the probability will never equal 1, there is not guarantee of a collision.
If input space is greater than 2^32, and the amount of memory is 2^{32} slots, then a collision must occur. With smaller amount of slots available, there is no guarantee a collision will occur.
With the above equations, however, a collision will occur in fifty percent of the time. Transforming this back to the Time-Space Tradeoff, the amount of slots is pertaining to memory, however, memory is measured as:

$$M = number of slots \times hash\ value\ size \times message\ size$$

For this simulation, the hash value size is 32 bits. For the following graph, assume the message size is 64 bits.
Assume that the available number of memory is .786 MB. Each slot takes up $32 + 64(96)$ bits of memory. That means we have space for 65536 slots. (2^{16})
Graph
x-axis: number of attempts
y-axis: probability a collision is achieved



The assumption is that the larger amount of memory, the less attempts, and therefore, less time, is needed to find a collision(on average). Below is a table:

Memory(slots)	Attempts to .5
2^1	1416810832.89572
2^2	743624987.364546
2^3	369244203.199411
2^4	183346835.618112
2^5	94906297.6242516
2^6	47453196.8121258
2^7	23238408.7301421
2^8	12453342.3007674
2^8	5730100.11306598
2^9	2905809.09126776
2^{10}	1441411.95018755
2^{11}	730292.27281694
2^{12}	373815.683671761
2^{13}	196429.248388087
2^{14}	124173.92091794
2^{15}	110453.810953575
2^{16}	153923.480229485

P1 is the probability a collision will be found before stage1 is over. As the amount of memory gets larger, P1 gets larger, and the average number of attempts necessary to find a collision gets smaller. This is the time-space tradeoff.