The Counterintuitive Mathematics of Coincidence: An Analysis of the Birthday Paradox and its Applications

Amelia Dignan[1], **Jack Dignan**[1]

Received May 3, 2026
Accepted August 16, 2026
Electronic access September 30, 2026

The Birthday Paradox is a classic veridical paradox, similar in nature to the Monty Hall problem and Simpson's Paradox. A veridical paradox is a situation that appears highly counterintuitive but is true. The Birthday Paradox provides an ideal example of the occasional sharp divergence between formal mathematics and human intuition. This paper derives an exact probability model for the Birthday Paradox using standard probability conventions. Recognizing that high factorial calculations can trigger computational limits, we also use a closed-form exponential approximation (derived from localized Taylor series expansions) to generate precise mathematical approximations. In order to validate the integrity and stability of the derived approximations, a computational experiment was performed using 100,000 discrete Monte Carlo trials per group size. This framework maps the scaling limits that drive collision vulnerability in cryptographic algorithms by modeling a hash function's bit-length as a finite space. This paper defines the parameters required to enforce collision resistance in modern digital signature schemes by mathematically demonstrating that the operational security boundary against arbitrary collision attacks scales with the square root of the total output space ($\approx 1.177\sqrt{N}$).

Keywords: Birthday Paradox, Cryptography, Hash Functions, Taylor Series, Birthday Attack, Probability Theory

Introduction

When it comes to estimating probabilities, people tend to default to simplified mental models, leading to frequent mistakes. There is perhaps no better evidence of this than trying to determine the chances of two events occurring simultaneously[1]. The Birthday Paradox, also known as the birthday problem, is an example of this mental shortcoming: it poses the question, "What is the probability of finding at least one similar pair having the same birthday in a group of n individuals?"[2]. Most people are surprised when told that in a group of only 23 individuals, the probability of at least two sharing a birthday is greater than 50%[3]. In this respect, the Birthday Paradox is similar in nature to The Monty Hall problem, which illustrates the counterintuitive nature of conditional probability[4], and Simpson's Paradox, which illustrates the effect of confounding variables on combined data sets[5].

The Birthday Paradox, the Monty Hall problem, and Simpson's Paradox are all examples of veridical, or truth-telling, paradoxes[6]. That is to say, mathematically validated facts that run counter to human intuition. The exact origin and history of the Birthday Paradox is unclear, but it is known to have been first circulated by number theorist Harold Davenport in 1927. However, formal mathematical modeling was later published by Richard Von Mises[7]. Conventional probability models demonstrate that given a group of 23 people, there is a 50.7% probability that at least two share a birthday[2], and that the probability of a shared birthday increases rapidly toward 99.9% as the group size approaches 70[8].

While interesting on its own, the math underlying the Birthday Paradox extends to real-life applications in areas such as cryptography and cybersecurity. Specifically, cryptographic hashing schemes[9] rely on the same mathematics to determine the bounds that establish collision vulnerabilities. Unfortunately, secondary sources often don't differentiate by type of attack and, as a result, obfuscate these limits. This paper uses the Birthday Paradox to address this issue by evaluating how derived mathematical approximations compare to actual security bounds. We evaluate the mathematical foundation of the Birthday Paradox and then use Taylor series approximations to calculate error bounds at various group sizes. Lastly, a Monte Carlo simulation using Python is utilized to test for computational accuracy. In doing so, this study also clears up common confusion in existing literature by showing how these mathematical properties apply to specific cyberattacks, specifically highlighting the difference between finding a random hash collision versus breaking a target preimage resistance.

Literature Review

Allocation and match problems have garnered the attention of mathematicians and statisticians for at least the last cen-

[1] *The Taft School, CT, USA*

tury. Following the initial work done by Von Mises on occupancy and partition probabilities[7], statisticians sought to push the boundaries of the uniform distribution assumption. Classical models assume there is an equal probability of selecting each day in a 365-day year[8], but demographic studies by Nunnikhoven[10] and Murphy[11] revealed seasonal variations in human births. In those studies, Nunnikhoven and Murphy proved that non-uniform distributions increase the odds of a match due to data clustering shrinking the effective sample space. As such, assuming a uniform distribution provides a conservative baseline for analysis.

In 1979, Ralph Merkle[9] and Gideon Yuval[12] independently discovered that the Birthday Paradox applied directly to cryptographic hash functions[13]. In "How to Swindle Rabin," Yuval demonstrates that the exponential scaling of pair combinations can be used to fake digital signatures on fraudulent files[12]. Yuval's revelation initiated a fundamental change in hash function design. Merkle[9] and Damgård[14] applied these exact limits when developing the Merkle-Damgård construction. By explicitly factoring in the square-root bound ($2^{L/2}$), their design became the standard foundation for modern hash functions.

Research in cryptography has increasingly focused on addressing the operational threats uncovered by the realities discovered by Yuval[12] and Merkle[9]. For example, the development of memory-hard sequential key derivation functions, such as scrypt[15] and Argon2[16], was driven by the necessity to protect finite target spaces against so-called brute-force attacks. This paper builds on these developments, combining the math behind the Birthday Paradox with modern cybersecurity risks.

Methodology

Our analysis utilizes an integrated analytical and computational research approach. Calculating the probability of a shared outcome within a finite system requires the use of fundamental probability axioms and the complement rule[17]. High factorial calculations can trigger computational limits, so we also use a closed-form exponential approximation (derived from localized Taylor series expansions)[18].

Our derivation relies on two primary assumptions taken from standard probability practices[8]:

1. **Independence:** Prior selections do not affect the probability of subsequent selections, meaning the selection of an item in a trial is independent and identically distributed (i.i.d.).

2. **Uniform Distribution:** Each item within the sample space has an equal likelihood of selection ($1/d$).

Real-world environments often violate uniform distribution assumptions, e.g., human birth seasonality, non-random cryptographic hash distributions. But, as noted in the Literature Review, clustering mathematically compresses the effective sample space and *increases* match likelihood. Thus, our derived model represents a conservative baseline for analysis.

To validate the integrity and error bounds of our derived approximation model, we performed Monte Carlo simulations. Python was used to simulate 100,000 discrete Monte Carlo trials per group size, providing an empirical baseline to isolate precision error profiles and confirm model convergence[19]. Group sizes were scaled incrementally from $n = 2$ to $n = 100$ within a finite target space of $D = 365$. The resulting baseline provided a dataset used to generate precision error profiles, measure absolute deviation, and confirm model convergence.

Theoretical Framework: Direct Probability vs. Axiomatic Probability

Probability is the long-term relative frequency of an event within an infinite number of trials[20]. Consequently, if we define event A, we can express the probability of A as:

$$P(A) = \lim_{n\to\infty} f_n(A)$$

where $f_n(A)$ is the relative frequency of A given n-trials.

For this study, event A can be defined as the occurrence of at least one pair of individuals sharing an identical attribute (a birth date) within a group of n people. Because "at least one shared match" includes potentially many, many different outcomes, e.g., P(exactly one pair), P(exactly two pairs), P(exactly three pairs)... and P(exactly n pairs), calculating the probability of at least one shared birthday directly is both labor-intensive and cumbersome[2]. Instead, using Kolmogorov's probability axioms[21] and the complement rule results in a far easier and considerably more eloquent process.

The complement rule of probability states that the probability of an event A happening is equal to one minus the probability of A not happening:

$$P(A) = 1 - P(A^c)$$

(Note: A^c is said to be the "complement of A.")

The complement of A can be interpreted as the scenario where no individuals share a birth date. This approach allows us to evaluate $P(A^c)$ using fundamental counting rules. We let $d = 365$, so the total available slots in the finite sample space is fixed. We assume entries are independent and identically distributed, so the total possible ways n individuals can have birthdays can be calculated using d^n, or 365^n.[8] This will serve as our denominator in our probability calculation.

	Person					
$n=$	**1**	birthday	**1**	=	365	possibilities
	2	" "	**2**	=	365	" "
	3	" "	**3**	=	365	" "
	.	.	.	.	.	" "
	.	.	.	.	.	" "
	.	.	.	.	.	" "
	n	" "	**n**	=	365	" "
Total			**n**	=	365^n	possibilities

For the numerator, sequential restrictions are applied to guarantee uniqueness. That is to say, the first individual has 365 available dates, the second individual has 364 remaining options, and the n-th individual has $365-n+1$ options. Multiplying these outcomes together yields the total number of distinct birthday permutations, expressed here using factorials:

$$365 \cdot 364 \cdot 363 \cdots (365-n+1) = \frac{365!}{(365-n)!} = {}_{365}P_n$$

Therefore, the exact probability $P(n)$ of at least one shared birthday is defined as:

$$P(n) = 1 - \frac{365!}{365^n(365-n)!} = 1 - \frac{{}_{365}P_n}{365^n}$$

As previously cited, Nunnikhoven[10] identified that human births don't follow a uniform distribution. To reiterate, though, any departure from a uniform distribution actually *increases* the probability of a matching pair due to the mathematics of clustering in a finite sample space[11]. As such, our derived uniform model represents a conservative baseline lower bound. Similarly, independent attribute allocation is assumed, thereby ignoring empirical dependencies like twin births. Lastly, leap years are not considered in order to maintain system constraints at a consistent $d = 365$. Given that the probability of a birth falling on February 29th is roughly $1/1461 \approx 0.068\%$, and that its statistical impact on the probability curve is therefore negligible, it was deemed not to have changed the system's behavior in any meaningful way.

The geometric behavior of $P(n)$ yields a classic "S-curve" that is characteristic of other veridical paradoxes: the probability curve does not scale linearly, but rather exhibits rapid growth driven by the combinatorial mathematics. The graph can be broken into three phases.

- **Phase 1: Slow Acceleration ($n = 2$ to $n = 10$):** The slope remains relatively flat. Because n is small, the number of available non-conflicting days dominates the number of possible pair combinations, keeping the probability of a match relatively low.
- **Phase 2: Inflection ($n = 11$ to $n = 45$):** This is the structural driver of the Birthday Paradox. As n exceeds 15, the binomial coefficient $\binom{n}{2}$ starts to scale quadratically, resulting in a curve that undergoes a massive acceleration in slope.
- **Phase 3: Asymptotic Convergence ($n > 45$):** The probability curve exhibits asymptotic behavior as it approaches the upper bound of 1.0. At $n = 70$, the probability reaches 99.9%.

At $n = 23$, the number of possible unique pairs within the room is calculated using the binomial coefficient:

$$\binom{23}{2} = \frac{23 \times 22}{2} = 253$$

Because the 253 unique pairs each have an independent $1/365$ chance of matching, the inflection threshold crosses 50.7297%. This demonstrates the underlying nature of the Birthday Paradox: intuition (mistakenly) focuses on individual sample count (n) when the actual probability is determined entirely by pair combinations ($\binom{n}{2}$).

Mathematical Approximations Using Infinite Series

Approximations are used in math for many reasons, but most frequently to circumvent computational complexity and bypass strict microprocessor limits[21]. For example, if we use the formula we derived:

$$\begin{aligned} P(\text{at least one shared birthday}) &= 1 - P(\text{no shared birthdays}) \\ &= 1 - \frac{365!}{(365-n)!365^n} \\ &= 1 - \frac{{}_{365}P_n}{365^n} \end{aligned}$$

For $n \geq 40$, a floating-point arithmetic overflow error is triggered on standard consumer microprocessors because the values exceed the maximum storable thresholds of standard 64-bit structures[22]. But we can use a relatively simple, closed-form exponential equation to solve for the group size, n, algebraically for a desired probability. First, let us consider the Taylor series used to approximate e^x:

$$e^x \approx 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^k}{k!} + \ldots$$

When $|x| < 1$ (so applicable to probabilities), the higher-powered terms become incredibly small as to be negligible, allowing us to approximate $e^x \approx 1 + x$, therefore $e^{-x} \approx 1 + (-x) \implies e^{-x} \approx 1 - x$.[23] Recall,

$$P(\text{no match}) = \left(\frac{365}{365}\right)\left(\frac{364}{365}\right)\left(\frac{363}{365}\right)\cdots\left(\frac{365-(n-1)}{365}\right)$$

for $k = 0, 1, 2, \ldots, n-1$. We can re-write each term in the form $\left(1-\frac{k}{365}\right)$, as follows:

$$P(\text{no match}) = 1\left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\cdots\left(1-\frac{n-1}{365}\right)$$

Now we apply the approximation $e^{-x} \approx 1-x$, where $x = \frac{k}{365}$ for $k = 0, 1, 2, \ldots, n-1$. Therefore,

$$P(\text{no match}) \approx e^{-0/365} \cdot e^{-1/365} \cdot e^{-2/365} \cdots e^{-(n-1)/365}$$

Given the product rule for exponents ($e^a e^b e^c = e^{a+b+c}$), we can re-write as follows:

$$P(\text{no match}) \approx e^{-(0/365+1/365+2/365+\cdots+(n-1)/365)}$$
$$P(\text{no match}) \approx e^{-\frac{1}{365}(1+2+\cdots+(n-1))}$$

Now note that $1+2+\cdots+(n-1)$ is the sum of the first $(n-1)$ positive integers. The sum of this finite series is found by $S_t = \frac{t(t+1)}{2}$. In our case, $t = (n-1)$, therefore:

$$\sum_{i=1}^{n-1} i == \frac{(n-1)((n-1)+1)}{2} = \frac{(n-1)n}{2} = \frac{n^2-n}{2}$$

So,

$$e^{-\frac{1}{365}(1+2+\cdots+(n-1))} \approx e^{-\frac{1}{365}\left(\frac{n^2-n}{2}\right)} \approx e^{-\frac{n^2-n}{730}}$$

Therefore, we can use $1-e^{-\frac{n^2-n}{730}}$ to approximate the probability of at least one shared birthday given a group of n people. That is,

$$P(\text{at least one shared birthday}) \approx 1-e^{-\frac{n^2-n}{730}}$$

The percent error peaks near the $n = 25$ threshold before trending towards zero (see Table 2). This behavior is guaranteed by Taylor's Theorem[23]. Because our substitution truncates the power series after the first linear term $(1-x)$, the truncation error of each linear term is dominated by the leading omitted remainder term, $\frac{x^2}{2}$, where $x = k/365$. So as n increases toward large (asymptotic) thresholds (e.g. $n \approx 200$), both the exact probability and the Taylor approximation converge at 1.000000, with the absolute error variance going to zero and thereby demonstrating the stability of the model.

Practical Applications in Cryptography

When von Mises first published on the Birthday Paradox, he could not have possibly understood or appreciated its future applications in the fields of computer science and cybersecurity. Nonetheless, almost 100 years later, the math at the core of the Birthday Paradox is critical in understanding and thwarting cyber threats commonly known as "Birthday Attacks."

Cryptographic Digest Frameworks and Signature Schemes

Within modern information security, a cryptographic hash function converts arbitrary variable-length input data into a fixed-size bitstring output, which serves as a sort of digital fingerprint, called a hash[24]. Hashes created by cryptographic hash functions have seven main characteristics, which work to ensure data integrity[25]:

1. Determinism: The same input will always generate the same output (hash).
2. Fixed output size: No matter the length of the input, the output will all have the same number of characters.
3. Avalanche effect: Any small change to an input (e.g., capitalizing a character) will result in a vastly different output.
4. Quick computation: Hash functions must be able to generate hashes rapidly to remain practical due to their common use.
5. Pre-image resistance: It is computationally infeasible to reverse a hash function and find the input if the output is given.
6. Second pre-image resistance: Given an input, it is practically impossible to find a different input that creates the exact same hash.
7. Collision resistance: Making it practically impossible to find any two inputs with the same output[26].

Critically, hash functions are not to be confused with digital signatures. A digital signature is a cryptographic security system used for proving three things about a digital message: authentication, integrity, and non-repudiation. Digital signatures use public key cryptography, in which each user has two keys: a private key, used to create the signature, and a public key, used to verify said signature[27]. The creator/sender of a digital file will run the digital file through a hash function, then process the hash with a signing algorithm using their private key. The recipient of the file will use the sender's public

Table 2: Statistical Validation: Exact vs. Taylor Approximation Probabilities

Group Size (n)	Exact Probability P(n)	Taylor Approximation $\hat{P}(n)$	Absolute Error	Percent Error (%)
2	0.002740	0.002736	-0.000004	-0.1369%
5	0.027136	0.027025	-0.000111	-0.4062%
10	0.116948	0.115991	-0.000957	-0.8187%
20	0.411438	0.405805	-0.005633	-1.3692%
23	0.507297	0.500002	-0.007295	-1.4381%
25	**0.568700**	**0.560412**	**-0.008288**	**-1.4573%**
30	0.706316	0.696320	-0.009996	-1.4153%
50	0.970374	0.965131	-0.005243	-0.5403%
70	0.999160	0.998662	-0.000498	-0.0498%
100	0.999999	0.999999	-0.000000	-0.0001%
200	1.000000	1.000000	0.000000	0.0000%

key to verify the signature, and is then able to verify the file. Hash functions are used as an initial step of the process, and are not the same as digital signatures.

The Formal Collision Scaling Boundary

Current computing powers render identifying a direct inversion matching a targeted hash out of reach. However, recognizing an arbitrary collision pair ($H(M_1) = H(M_2)$ where $M_1 \neq M_2$) is a substantially easier goal, due to the scaling limitations of the Birthday Paradox[28]. To obtain the exact cryptographic vulnerability framework, let $N = 2^L$ represent the total output space of an L-bit hash function. Setting the intended success threshold to $P(\text{collision}) = 0.5$, we map the mathematical transition using the exponential approximation derived in Equation 5.

$$1 - e^{-\frac{n^2-n}{2N}} = 0.5$$

By isolating the exponent, this simplifies to:

$$e^{-\frac{n^2}{2N}} \approx 0.5$$

Taking the natural logarithm of both sides gives:

$$-\frac{n^2}{2N} \approx \ln(0.5) = -\ln(2)$$

Finally, solving for the number of inputs n yields:

$$n \approx \sqrt{2(\ln 2)N} \approx 1.1774\sqrt{N} = 1.1774 \times 2^{L/2}$$

This derivation reveals that the operational security boundary of any standard hash algorithm scales not with its full bit-length L, but with its square root exponent $2^{L/2}$.[9] Consequently, a legacy 128-bit hash algorithm (such as MD5) yields a real-world collision resistance boundary of only 2^{64} execution operations—a constraint easily broken by modern parallel computing clusters[29]. This is why modern enterprise infrastructure mandates SHA-256 or SHA-3, where the 2^{128} collision boundary remains safe from classical computing attacks[26].

Reframing Exploitation Models: Collisions vs. Preimages

A collision attack relies on finding collision pairs, two unique inputs with the same hash. In a classic digital signature forgery, a hacker will create two contracts: a safe, legally binding one, and a malicious one. Then, by manipulating small details within each contract, the hacker creates many different variations of each document. The numerous contracts are hashed and compared until a match is found between a legal contract and a fraudulent one. Once a target digitally signs the legal contract, the hacker can use that digital signature on the fraudulent document, deceiving the system into thinking it is legitimate. This sort of attack is defended against most readily by increasing the hash lengths. Hackers rely on older models, such as 128-bit MD5 or 160-bit SHA-1, to execute Birthday Attacks. Modern algorithms featuring 256-bit or larger outputs make finding matching hashes nearly impossible with current computing power, due to the sheer amount of attempts needed.

Conversely, a Birthday Attack is not utilized in authentication bypass scenarios, such as stealing passwords from leaked database hashes. In those situations, the hacker is attempting to find an input that matches a specific hash, rather than two inputs that have the same hash. This sort of attack is known as a First Preimage Attack, and (within modern hash functions) relies entirely on brute force. Hackers will attempt to run common passwords through the hash function until they find a match with their hash. However, this sort of attack is computationally infeasible with modern hash functions given the immense required processing power.

Conclusion

The power of probability and the shortcomings of intuition are perhaps no better exemplified by the counterintuitive nature of the Birthday Paradox. It also demonstrates the usefulness and eloquence of axiomatic probability. Furthermore, our computational experiment confirms that truncating the Taylor series expansion yields stable, closed-form exponential approximations that effectively address float-overflow constraints without introducing significant error variance.

It is important to note three main limitations:

1) **The Uniformity Distribution and Independence Assumption Gap:** Our derived model relies on independent and identically distributed sample spaces. These assumptions are unlikely to hold in many real-world environments. For example, demographic data indicate a month like August consistently records more births than a month like February. This type of non-uniform clustering compresses the effective sample space and increases the probability of a match.

2) **Non-Random Adversarial Exploitation:** Our convergence analysis and Monte Carlo simulations assume inactive, random collision pairings. In real-world cryptographic attacks, inputs are often not random. Strategic cybercriminals utilize targeted attacks, such as searching for personal information on the internet, then incorporating that personal information into their cyber attacks. Because hostile inputs are skewed by intent rather than random chance, practical security boundaries may fall sooner than our theoretical thresholds suggest.

3) **The Computing Optimization Bound:** This paper uses the 64-bit floating-point overflow on standard microprocessors to justify our Taylor series approximation. Increases in modern compute power are rendering this overflow hurdle increasingly obsolete. Modern computers have developed precise mathematical algorithms capable of computing immense calculations without overwhelming their systems. As a result, our Taylor series approximation should be viewed as a quick and easy shortcut, rather than a necessity.

The Birthday Paradox exposes an inherent and hidden weakness within modern cryptography; collision immunity is mathematically impossible due to the pigeonhole principle. The threat of a collision attack scales proportionally with the square root of its output space ($\approx 1.177\sqrt{N}$), effectively halving the expected bit security of standard hashing schemes. This places a rigid constraint on modern computing platforms: to maintain security integrity, algorithms must utilize massive bit-lengths (such as SHA-256) to ensure that the Birthday Attack remains computationally nearly impossible.

Declaration of Generative AI in Research

During the preparation of this study, ChatGPT (OpenAI) was used to write, troubleshoot, and debug the Python code used for our Monte Carlo simulations. The authors manually reviewed, executed, and verified all code outputs to ensure technical accuracy, logical consistency, and reproducible results. The authors maintain full responsibility for the data and final content presented in this manuscript.

Author Contributions

Both authors contributed to the study conception and design. The Literature Review, Practical Applications in Cryptography, and Conclusion sections were written by A.D. Mathematical Approximations Using Infinite Series and Python coding were conducted by J.D. Both authors contributed to the Methodology and Theoretical Framework sections. Both authors also commented on previous versions and were involved in subsequent revisions. Lastly, both authors read and approved the final manuscript.

References

1 D. Kahneman and A. Tversky, *Subjective probability: A judgment of representativeness*, 1972, 10.1016/0010-0285(72)90016-3.

2 A. DasGupta, *The matching, birthday and the strong birthday problem: A contemporary review*, 2005, 10.1016/j.jspi.2003.11.015.

3 P. Diaconis and F. Mosteller, *Methods for studying coincidences*, 1989, 10.1080/01621459.1989.10478847.

4 S. Selvin, *A problem in probability [Letter to the editor]*, 1975, 10.2307/2683689.

5 C. H. Wagner, *Simpson's paradox in real life*, 1982, 10.1080/00031305.1982.10482778.

6 W. V. Quine, *The Ways of Paradox and Other Essays*, 1976.

7 R. Von Mises, *Über Aufteilungs- und Besetzungsaufgaben*, 1939.

8 W. Feller, *An Introduction to Probability Theory and Its Applications*, 1968.

9 R. C. Merkle, *One way hash functions and DES*, 1989, 10.1007/0-387-34805-0_40.

10 T. S. Nunnikhoven, *A birthday problem with non-uniform birthdays*, 1992, 10.2307/2685062.

11 R. Murphy, *Nonuniform birthday distributions and the birthday paradox*, 1993, 10.2307/3214716.

12 G. Yuval, *How to swindle a digital signature*, 1979.

13 S. Landau, *Find me a hash*, 2006.

14 I. Damgård, *A design principle for hash functions*, 1989, 10.1007/0-387-34805-0_39.

15 C. Percival, *Stronger key derivation via sequential memory-hard functions*, 2009.

16 A. Biryukov, D. Dinu and D. Khovratovich, *Argon2: New generation of memory-hard functions for password hashing and key derivation*, 2016, 10.1109/EuroSP.2016.31.

17 P. S. Laplace, *Théorie Analytique des Probabilités*, 1812.

18 D. E. Knuth, *The Art of Computer Programming: Seminumerical Algorithms*, 1997.

19 N. Metropolis and S. Ulam, *The Monte Carlo method*, 1949, 10.1080/01621459.1949.10483310.
20 R. Von Mises, *Probability, Statistics and Truth*, 1928.
21 A. N. Kolmogorov, *Foundations of the Theory of Probability*, 1933.
22 IEEE Computer Society, *IEEE Standard for Floating-Point Arithmetic (IEEE Std 754-2008)*, 2008, 10.1109/IEEESTD.2008.4610935.
23 K. Miranda and M. Sullivan, *Calculus*, 2023.
24 A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, 1996.
25 W. Stallings, *Cryptography and Network Security: Principles and Practice*, 2017.
26 National Institute of Standards and Technology, *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (FIPS PUB 202)*, 2015, 10.6028/NIST.FIPS.202.
27 R. L. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, 1978, 10.1145/359340.359342.
28 M. Bellare and T. Kohno, *Hash function balance and its impact on birthday attacks*, 2004, 10.1007/978-3-540-24676-3_24.
29 X. Wang and H. Yu, *How to break MD5 and other hash functions*, 2005, 10.1007/11426639_2.

Appendix: Python Script

```
import numpy as np
import math

def run_birthday_simulation(group_size, total_slots=365, trials=100000):
    """
    Empirically simulates the paradox using random uniform integer sampling
    to validate the probability of a shared outcome within a finite space.
    """
    collision_count = 0
    for _ in range(trials):
        # Generate uniform random distribution vectors
        sample_space = np.random.randint(0, total_slots, size=group_size)
        if len(sample_space) != len(set(sample_space)):
            collision_count += 1
    return collision_count / trials

def calculate_exact_and_approx(n, N=365):
    """Computes exact combinatorial logic vs Taylor exponential models."""
    if n > N:
        exact = 1.0
    else:
        exact = 1.0 - math.prod([(N - i) / N for i in range(n)])
    approx = 1.0 - math.exp(-(n**2 - n) / (2 * N))
    return exact, approx

# Target evaluation points requested by peer review
test_points = [10, 23, 50]

print(f"{'n':<6}{'Empirical P(n)':<18}{'Exact P(n)':<15}{'Taylor Approx':<15}")
print("-" * 60)
for n in test_points:
    empirical = run_birthday_simulation(n)
    exact, approx = calculate_exact_and_approx(n)
    print(f"{n:<6}{empirical:<18.6f}{exact:<15.6f}{approx:<15.6f}")
```