electronics

MDPI

Article

Small Prime Divisors Attack and Countermeasure against the RSA-OTP Algorithm

Szymon Sarna [1,2,*] and Robert Czerwinski [2]

[1] Digital Core Design, 41-902 Bytom, Poland
[2] Department of Digital Systems, Silesian University of Technology, 44-100 Gliwice, Poland; rczerwinski@polsl.pl
* Correspondence: szymon.sarna@polsl.pl

Abstract: One-time password algorithms are widely used in digital services to improve security. However, many such solutions use a constant secret key to encrypt (process) one-time plaintexts. A paradigm shift from constant to one-time keys could introduce tangible benefits to the application security field. This paper analyzes a one-time password concept for the Rivest–Shamir–Adleman algorithm, in which each key element is hidden, and the value of the modulus is changed after each encryption attempt. The difference between successive moduli is exchanged between communication sides via an unsecure channel. Analysis shows that such an approach is not secure. Moreover, determining the one-time password element (Rivest–Shamir–Adleman modulus) can be straightforward. A countermeasure for the analyzed algorithm is proposed.

Keywords: RSA; OTP; one-time password; RSA-OTP; small prime divisors attack; cryptography; encryption

Citation: Sarna, S.; Czerwinski, R. Small Prime Divisors Attack and Countermeasure against the RSA-OTP Algorithm. *Electronics* **2022**, *11*, 95. https://doi.org/10.3390/electronics11010095

Academic Editors: Xiaohong Jiang, Yulong Shen, Yuanyu Zhang and Tarik Taleb

Received: 7 November 2021
Accepted: 24 December 2021
Published: 28 December 2021

1. Introduction

Cryptography is the basis of modern secure communication. Cryptographic algorithms are very important for the security of data or information [1]. Commonly used algorithms such as the Advanced Encryption Standard (AES) and the Secure Hash Algorithm 2 (SHA-2) are often assumed to be unbreakable. However, the predecessors of these algorithms typically experienced decreased levels of security throughout their lifetimes. Hence, despite modern algorithms appearing secure, increasingly safe solutions must be sought [2–4].

Commonly used algorithms are often combined with an authentication process such as one-time password (OTP) to enhance communication security. The most popular OTP implementations, known as HOTP algorithms [5], are derived from hash-based message authentication code (HMAC) [6] algorithms. Given that HMAC algorithms use hash functions as symmetric keys, they represent ideal solutions for challenge-response authentication. However, the algorithms are effectively one-way and so cannot be used directly to encrypt data. This problem is not shared by AES-based OTP authentication algorithms (AES-OTP) [7]. These two examples, despite differing in approach and algorithm utilization, both use an OTP mechanism based on non-repeating challenges with a constant cryptographic key.

Jabłoński and Wójtowicz [8] propose a novel use of the Rivest–Shamir–Adleman (RSA) [9] algorithm in the context of OTP (RSA-OTP): following each sign or encryption operation, the RSA modulus n_i is replaced by a newly generated value n_{i+1}. This approach ensures the secrecy of the initial key elements and their secure exchange between two communicating parties. Although this OTP-based approach no longer uses a public key, an important RSA attribute, it allows using shorter keys.

The primary advantage of RSA-OTP over AES-OTP or HOPT is a kind of one-time pad structure: each message is encrypted with a new random key which can be used for native cloning detection. A further advantage is the asymmetric nature of RSA and its

mathematical simplicity. Research throughout the past four decades has shown that the primary threat to RSA is large integer factorization, a result of public knowledge of used moduli. Some research [10,11] additionally shows that the implementation method of RSA can produce unintended weaknesses. However, in the case of secret exponents and moduli, many public key RSA threats are no longer relevant.

The novel contributions of this paper are a small prime divisors attack against the RSA-OTP algorithm [8] and a countermeasure using XOR operation.

2. The RSA Algorithm

Many applications require the secure communication or storage of data. The RSA algorithm is a commonly used solution in various types of security applications. Despite the difficulty in breaking RSA keys, RSA algorithms still suffer varied attacks [12,13].

As an asymmetric encryption algorithm, RSA is widely deployed in public key cryptosystems (public key cryptography). A plaintext is encrypted by a public key and the resultant ciphertext is transmitted to the receiver where it can be decrypted by a private key. Encryption security is contingent on the difficulty of factoring the product of two large prime numbers [14].

To generate a private and public key pair, two large prime numbers p and q are randomly chosen. They are used to generate a semiprime n, where:

$$n = p \cdot q, \tag{1}$$

and Euler's totient function $\varphi(n)$, where:

$$\varphi(n) = (p-1) \cdot (q-1). \tag{2}$$

The public key (n, e) is a pair, where e is an integer. Additionally, e is not a factor of $\mod n$, and $1 < e < \varphi(n)$. The private key (n, d) is a pair, where d is the modular multiplicative inverse of e modulo $\varphi(n)$:

$$d \equiv e^{-1} \mod \varphi(n). \tag{3}$$

The public key encrypts a plaintext x into a ciphertext y:

$$y \equiv x^e \mod n. \tag{4}$$

To recover the original message, the corresponding private key decrypts the ciphertext:

$$x \equiv y^d \mod n. \tag{5}$$

Example 1. *Let us choose two prime numbers $p = 11$, $q = 17$ and compute $n = 187$ by means of (1). The Euler's totient (2) is $\varphi(n) = 160$. Let us choose e such that $1 < e < \varphi(n)$ and e and $\varphi(n)$ are coprime: $e = 7$. Let us compute a value for d (3): 23. The public key is $(e, n) => (7, 187)$, while the private key is $(d, n) => (23, 187)$. Let us encrypt letter A <ASCII 65>. The encryption (4) of $x = 65$ is $y = 142$ and the decryption (5) x of y is 65.*

3. The RSA-OTP Concept

The RSA-OTP algorithm also uses asymmetric keys e and d_i. Following each encrypted transaction, a new modulus generation process is initiated by one of the communicating parties. The party is hence the owner of key d_i. A differential value Δn_i is created from the previous modulus n_{i-1} and the new modulus n_i:

$$\Delta n_i = n_i - n_{i-1}. \tag{6}$$

This value is then transmitted to the second party: the owner of key e. As claimed by Jabłoński and Wójtowicz [8], Δn_i can be exchanged across a public, unsecure channel. According to the authors, such a communication method guarantees an "unconditional

security level". Moreover, the modulus length at even 128 bits should be sufficiently secure for certain applications. RSA-OTP scheme is presented in Figure 1.

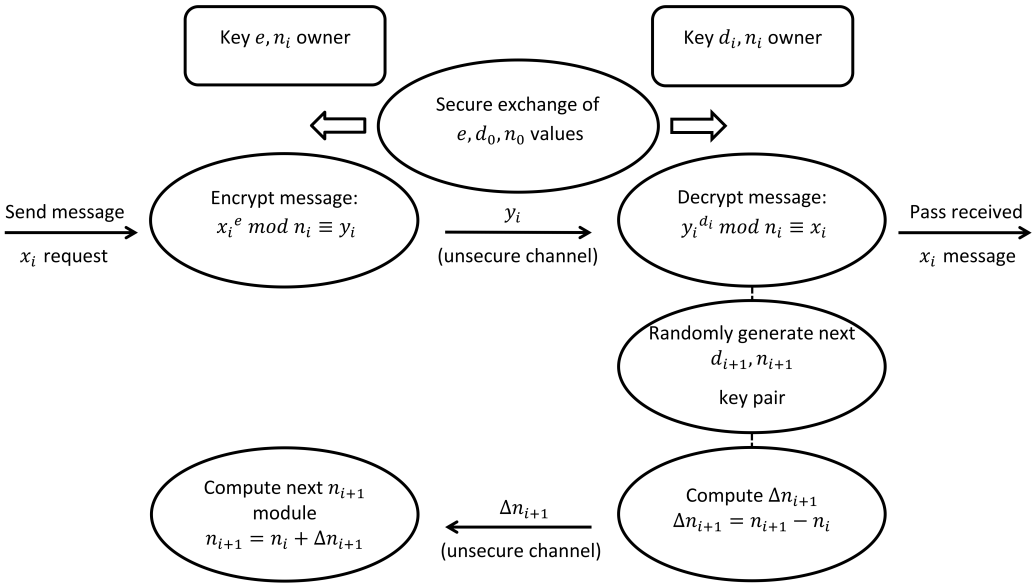

Figure 1. RSA-OTP scheme.

Assuming the privacy of each modulus n_i, 128-bit RSA-OTP can be secure under several conditions. However, given that the Δn_i values are closely related to the n_i values, the unsecure exchange of Δn_i values raises serious concerns regarding unintended n_i leakage. These concerns precipitate a deeper analysis of the RSA-OTP approach.

Consider the dependency between modulus n and ciphertext y within RSA, as described in Equation (4). Given that y is always smaller than n, y represents the lower bound of the range which can contain a modulus. The upper bound is the product of the two largest primes for which the bit length is equal to the maximum modulus size (e.g., 128-bit). From this, an attacker observing RSA-OTP communication can attempt to estimate the boundaries of the current moduli via Algorithm 1.

Algorithm 1 k-bit RSA-OTP modulus bounding

$m_{min} = 2^{k-1}$
$m_{max} = 2^k - 1$
$m_l = m_{min}$
for each $(y_i, \Delta n_{i+1}) \Leftarrow RSA\text{-}OTP_{TRX}()$ **do**
  if $y_i > m_l$ **then**
    $m_l \leftarrow y_i$
  end if
  if $m_l + \Delta n_{i+1} < m_{min}$ **then**
    $m_l \leftarrow m_{min}$
  else
    $m_l \leftarrow m_l + \Delta n_{i+1}$
  end if
  if $m_l > m_{max}$ **then**
    $m_l \leftarrow m_{max}$
  end if
end for

During empirical tests with 128-bit moduli, observations show that after $5 \cdot 10^6$ transactions in which ciphertext y_i and differential value Δn_i are exchanged, at least twenty of the most significant bits of m_l and n_i match. Following this, to further reduce the security

of modulus n_i, the condition in which many of the most significant bits of m_l are set must be detected. When this occurs, the number of possible modulus combinations decreases substantially, as shown in Table 1.

Table 1. Summary of RSA-OTP moduli security as a function of the number of known most significant bits (MSBs) of 128-bit moduli.

Number of Modulus MSBs Set	Security Level in Bits	Approximate Number of Observed RSA-OTP Transactions After Which, with Probability $p = \frac{1}{2}$, Some Modulus Will Have a Distinct Number of MSBs Set
1	115	1
2	113	2
3	111	8
⋮	⋮	⋮
18	81	2^{33}
19	79	2^{35}
20	77	2^{37}
⋮	⋮	⋮

The results presented in the first two columns of Table 1 are prime number approximations generated by

$$\pi(x) \approx \frac{x}{log(x) - 1}. \tag{7}$$

Consider the smallest 128-bit number t that has its s most significant bits set (e.g., for $s = 3 : t = (111000...000)_2$). The division of t by the largest 64-bit prime number p_{max} produces a lower bound p_{min} of possible prime values that can generate t or greater. Determining these prime values allows the estimation of prime numbers in a given range as follows:

$$primeNo \approx \pi(p_{max}) - \pi(p_{min}). \tag{8}$$

The number of unique combinations $\binom{primeNo}{2}$ represents the estimated number of possible modulus values, as displayed in the third column of Table 1. Among these combinations exist many values that do not produce sufficient "large" moduli. Such values can be justifiably included given that their existence can only be verified following the multiplication of two primes larger than p_{min}. To summarize, RSA-OTP modulus leakage occurs via the following process:

- Observation of approximately consecutive random $5 \cdot 10^6$ RSA-OTP transactions, using Algorithm 1, determines at least 20 most significant bits of each modulus.
- Following this, the condition being reached that many of the most significant bits of m_l are set brings a substantial decrease in the number of possible primes that can generate modulus n_i.

Empirical tests over $32 \cdot 10^6$ RSA-OTP 128-bit transactions bounded the number of possible prime combinations that could generate the sought moduli to 91–84 bits. While this represents substantial progress toward discovering the RSA-OTP moduli, their final designation remains a computationally demanding task. Moreover, a trivial countermeasure can be applied to prevent such reduction in possible combinations of RSA-OTP moduli. This entails the verification of generated moduli and the rejection of those for which many most significant bits are set. The maximum modulus value can be limited to $(BFF...FFF)_{16}$. Additionally, the extension of moduli to 256-bit should make this type of attack infeasible.

4. Small Prime Divisors Attack

The RSA-OTP attack method presented in the previous section degrades security but is somewhat impractical. This section presents an alternative attack method. Three lemmas lay the framework.

Lemma 1. *If a value m_0 is a sought RSA-OTP modulus n_0, then m_0 possesses only two divisors with bit length approximately half that of m_0. Similarly, each of the successive moduli $m_1, m_2, ..., m_{k-2}, m_{k-1}$, computed by the addition of RSA-OTP Δn_i values, possess only two divisors with bit length approximately half that of the corresponding modulus.*

Lemma 2. *For a prime number p and a k-element set of random numbers $R = \{r_0, r_1, ..., r_{k-2}, r_{k-1}\}$, such that $\forall r_i \in R : r_i \gg p$, the probability that p is not a divisor of any r_i is $\left(\frac{p-1}{p}\right)^k$.*

Lemma 3. *Consider a set of "candidates" for RSA-OTP moduli $M = \{m_0, m_1, ..., m_{k-2}, m_{k-1}\}$ which are determined using $m_i - m_{i-1} = \Delta n_i$. The RSA-OTP moduli n_i are generated in a random manner such that the $\Delta n_i = n_{i+1} - n_i$ are also random. Hence, a k-element set of candidates M can be also considered random, and when it does not contain real moduli ($m_i \neq n_i$), then with probability $1 - \left(\frac{p-1}{p}\right)^k$ at least one of them will be divisible by the selected, smaller prime number p.*

This attack is derived from the fact that prime numbers, or large composite numbers such as RSA moduli, do not have small divisors (Lemma 1). Hence, such numbers can be "reconstructed" using equations for values that are not divisible by a set of small primes. The divisibility of integers by prime numbers is cyclic. Primes $p_0 = 2$ and $p_1 = 3$ have a cycle of $6 = lcm(2, 3)$. Primes p_0, p_1, and p_2 have a cycle of $30 = lcm(2, 3, 5)$. In this manner, any value s which is non-divisible by the set of the first k primes can be described as:

$$s = t + p_0 \cdot p_1 \cdot ...p_{k-2} \cdot p_{k-1} \cdot i, \tag{9}$$

where the *offset* t is an odd natural number and i is the largest natural number such that $p_0 \cdot p_1 \cdot ...p_{k-2} \cdot p_{k-1} \cdot i < s$. Hence, there exists only one possible t which is always smaller than $p_0 \cdot p_1 \cdot ...p_{k-2} \cdot p_{k-1}$. Via this process, discovery of RSA moduli is undertaken by considering only odd values. As such, it holds that for every RSA-OTP modulus.

$$n = 1 + 2 \cdot i_0. \tag{10}$$

None of the practically used RSA-OTP moduli are divisible by 3, which, in conjunction with Equation (10), results in three possible offsets to describe such moduli:

$$n = 1 + 2 \cdot 3 \cdot i_1, \tag{11}$$

$$n = 3 + 2 \cdot 3 \cdot i_1, \tag{12}$$

$$n = 5 + 2 \cdot 3 \cdot i_1. \tag{13}$$

Example 2. *Let us assume that $n = 13 \cdot 17 = 221$. It can be observed that Equation (13) holds. For the next prime, 5, noting that each offset differs by $6 = 2 \cdot 3 = p_0 \cdot p_1 \cdot ... \cdot p_{k-3} \cdot p_{k-2}$, the following equations must be verified:*

$$n = 5 + 2 \cdot 3 \cdot 5 \cdot i_2, \tag{14}$$

$$n = 11 + 2 \cdot 3 \cdot 5 \cdot i_2, \tag{15}$$

$$n = 17 + 2 \cdot 3 \cdot 5 \cdot i_2, \tag{16}$$

$$n = 23 + 2 \cdot 3 \cdot 5 \cdot i_2, \tag{17}$$

$$n = 29 + 2 \cdot 3 \cdot 5 \cdot i_2. \tag{18}$$

Of these, Equation (15) *holds. Hence, for the following prime,* 7*, the modulus* n *is described by an offset* 11*:*

$$n = 11 + 2 \cdot 3 \cdot 5 \cdot 7 \cdot i_3. \tag{19}$$

As $lcm(2,3,5,7,11) > n$ *(where: lcm is least common multiple), Equation* (19) *must determine the correct value of* n*, with* $i_3 = 1$*.*

Table 2 summarizes the attack. It presents the number of primes required to precisely determine moduli of set binary sizes. The attack is surprisingly effective for even 4096-bit moduli with a computational complexity on the order of 32 bits. The requirement to observe a large set of consecutive Δn_i values presents the only drawback. This follows from Lemma 2: to achieve a high probability of success, the examined set must be sufficiently large so as not to verify an offset as a false positive (i.e., non-divisible by the selected prime number).

Table 2. hlSummary of the RSA-OTP small prime division attack.

k-th Prime Number	p_{k-1} Prime Number	Bit Length of RSA-OTP Modulus	Number of Observed Δn_i Values for Attack Error Probability $e = 10^{-15}$	Estimated Attack Complexity (Bits)
26	101	128	3472	20
44	193	256	6649	22
76	383	512	13,212	23
132	743	1024	25,646	26
234	1481	2048	51,135	29
419	2897	4096	100,042	32

For each subsequent small prime, the number of equations increases. As such, the discovery of the correct offset for primes larger than two appears difficult. However, the task is made considerably easier in the case of RSA-OTP by the observation of a sufficiently large set of consecutive public Δn_i values. The complete process is presented in Algorithm 2, which demonstrates the determination of the correct offset using Lemma 3. Upon validation of the correct offset, none of the checked values are divisible by the selected prime (line 18 in Algorithm 2). In another case, with probability $\left(\frac{p-1}{p}\right)^k$ (Lemma 2) some value will be divisible by the selected prime.

The designation of the RSA-OTP secret moduli makes treating them as a one-time element meaningless but does not lead to a complete breakdown of the RSA-OTP scheme: all time exponents e and d_i remain unknown. However, in specific scenarios, especially for short (128/256 bit) version of RSA-OTP, it may seriously affect the secrecy of encrypted data.

Algorithm 2 Small prime divisors attack

Require:
$\{bitLen, k, \{k \text{ depends on } bitLen, \text{see Table 2}\}$
$\{\Delta n_1, \Delta n_2, ..., \Delta n_{k-2}, \Delta n_{k-1}\}$

Ensure: $\{n_0, n_1, ..., n_{k-2}, n_{k-1}\}$
$lcmCycle \leftarrow \{1, 1\}$
$prime \leftarrow 2$
$m_0 \leftarrow 1$
for $i = 1$ **to** $k - 1$ **do**
  $m_i \leftarrow m_{i-1} + \Delta n_i$
end for
while $bitLength(lcmCycle_0) < bitLen$ **do**
  $lcmCycle_0 \leftarrow lcmCycle_1$
  $lcmCycle_1 \leftarrow lcmCycle_1 \cdot prime$
  for $i = 0$ **to** $prime - 1$ **do**
    $offsetFit \leftarrow True$
    $\Delta offset \leftarrow i \cdot lcmCycle_0 \mod lcmCycle_1$
    for $j = 0$ **to** $k - 1$ **do**
      if $m_j + \Delta offset \mod prime \equiv 0$ **then**
        $offsetFit \leftarrow False$
        break
      end if
    end for
    if $offsetFit = True$ **then**
      $prime \leftarrow nextPrime(prime)$
      for $j = 0$ **to** $k - 1$ **do**
        $m_j \leftarrow m_j + \Delta offset$
      end for
      break
    end if
  end for
end while

return m

5. Countermeasures

The primary conclusion to be drawn from the presented attack is the need for secure, encrypted exchange of RSA-OTP Δn_i values. Multiple possible solutions exist, the most straightforward of which is an AES encryption algorithm. However, this approach would significantly increase the complexity of the solution. On the other hand, the Δn_i values look pseudo-random to an external observer, as shown in Figure 2. This bitmap, containing 256 randomly generated consecutive Δn_i values exchanged between communication parties, looks like (except for the few most significant bits) white noise.

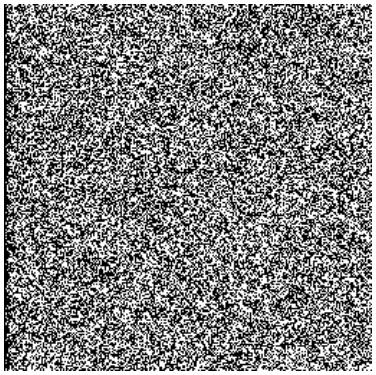

Figure 2. Example bitmap of 256 Δn_i values from following 256-bit RSA-OTP random transactions (one value per row). (1) $0x27CBB0EADA66F8C1A8FA444A1D738E4E124F38F0A0C5CD10110066E9724DAAF5$*. (2) $0x5B8D5B5B9DE5E771319CEF62D99CF5169B994F7DD34CDBC2F45462B797D7E9D8$* ... (256) $0x5D38FBAFB2A93048AA4850CE4B70E3A26E8C0BD4C130FF5F4F61354F4A7371B6$*. * The least significant bit of each value is a sign bit (1: negative, 0: positive).

Given the random nature of the Δn_i values, a more effective approach could be implemented. Consider a basic XOR operation between consecutive differential values:

$$\Delta encn_i = \Delta n_i \oplus \Delta n_{i-1}. \tag{20}$$

Assuming Δn_0 is a secret, random value securely exchanged between both communication sides, this approach nullifies the presented attack because an attacker cannot determine Δn_i from $\Delta encn_i$. As shown in Figure 3 $\Delta encn_i$ values except for those most significant bits are approximately white noise. The figure presents a series of 256 $\Delta encn_i$ transactions corresponded to ones in Figure 2 and Δn_0: $0xFB945A7D42485E3A0A5D2F346BAA9455E3E70682C2094CAC629F6FBED82C07CD$.

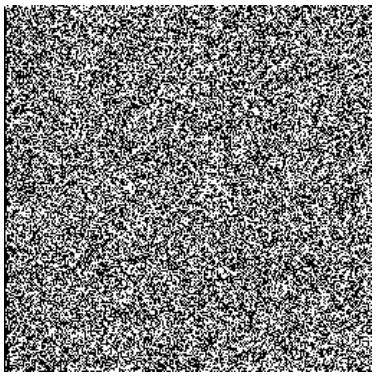

Figure 3. Example bitmap of 256 $\Delta encn_i$ values from following 256-bit RSA-OTP random transactions (one value per row). (1) *0xDC5FEA97982EA6FBA2A76B7E76D91A1BF1A83E7262CC81BC739F0957AA61AD38*. (2) *0x7C46EBB147831FB09966AB28C4EF7B5889D6778D738916D2E554045EE59A432C* ... (256) *0x177A17001C0DAC110B7F4EDF42428ABC497E54C5684E0A737C7EC7BE2FD6DC08*.

The improved RSA-OTP scheme is presented in Figure 4.

The principle of operation (20) is similar to that of the Vernam cipher, because each Δn_i is derived from two randomly generated values (6) and then $\Delta encn_i$ is a XOR result of two such consecutive values; hence, an attacker observing such communication cannot revert original value from it. As in the case of the Vernam cipher, achieving secrecy of exchanged data does not guarantee its integrity and authenticity; however, the simplicity of the mechanism increases the security of the OTP algorithm. Depending on the application, algorithms such as HMAC or Poly1305 can assure the integrity and authenticity of the $\Delta encn_i$ approach.

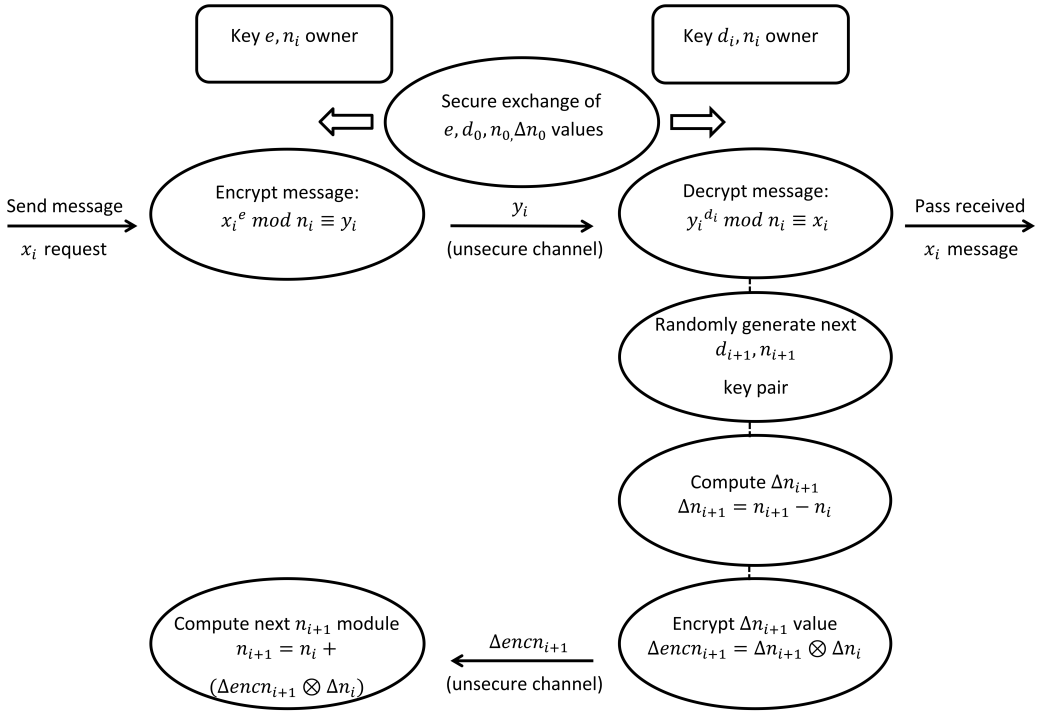

Figure 4. Improved RSA-OTP scheme.

6. Conclusions

The paradigm shift from constant to one-time keys is an interesting direction in the development of secure applications and services. Such an approach can potentially solve many problems regarding recovery following a system being compromised and resistance against side channel attacks. However, the case of RSA-OTP shows that exact RSA-OTP moduli delta values exchanged via insecure channels can be totally compromised, invalidating their classification as an OTP element. This is caused by the particular nature of those values: they are products of relatively large prime numbers which have no small divisors. Fortunately, despite the success of the small prime divisors attack, basic encryption via XORing consecutive differential values should prove to be a sufficient countermeasure.

Author Contributions: Conceptualization, S.S.; methodology, S.S.; software, S.S.; validation, S.S.; formal analysis, S.S.; investigation, S.S. and R.C.; resources, S.S. and R.C.; data curation, S.S.; writing—original draft preparation, S.S. and R.C.; supervision, S.S. funding acquisition, R.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Polish Ministry of Science and Higher Education funding for statutory activities.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RSA	Rivest–Shamir–Adleman
OTP	One-Time Password
AES	Advanced Encryption Standard
HMAC	Hash-based Message Authentication Code
HOTP	HMAC OTP
XOR	Exclusive or

References

1. Obaid, T. Study A Public Key in RSA Algorithm. *Eur. J. Eng. Technol. Res.* **2020**, *5*, 395–398. [CrossRef]
2. Alkim, E.; Ducas, L.; Pöppelmann, T.; Schwabe, P. Post-quantum key exchange—A new hope. In Proceedings of the 25th USENIX Security Symposium, Austin, TX, USA, 10–12 August 2016.
3. Rajasekar, P.; Premalatha, J.; Sathya, K. Multi-factor signcryption scheme for secure authentication using hyper elliptic curve cryptography and bio-hash function. *Bull. Pol. Acad. Sci. Tech. Sci.* **2020**, *68*, 923–935. [CrossRef]
4. Nastase, L. Security in the Internet of Things: A Survey on Application Layer Protocols. In Proceedings of the 21st International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 29–31 May 2017; pp. 659–666.
5. M'Raihi, D.; Bellare, M.; Hoornaert, F.; Naccache, D.; Ranen, O. *RFC4226: HOTP: An HMAC-Based One-Time Password Algorithm*; RFC, Ed.; Internet Engineering Task Force (IETF): Fremont, CA, USA, 2005.
6. Krawczyk, H.; Bellare, M.; Canetti, R. *RFC2104: HMAC: Keyed-Hashing for Message Authentication*; RFC, Ed.; Internet Engineering Task Force (IETF): Fremont, CA, USA, 1997.
7. Yubico, Yubico-OTP. Available online: https://developers.yubico.com/OTP/OTPs_Explained.html (accessed on 10 November 2021).
8. Jabłoński, J.; Wójtowicz, M. Unconditionally Secure Cryptographic System (Bezwarunkowo bezpieczny system kryptograficzny). *Logistyka* **2014**, *12*, 611–616. (In Polish)
9. Rivest, R.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
10. Kocher, P. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Proceedings of the Advances in Cryptology—CRYPTO'96, Santa Barbara, CA, USA, 18–22 August 1996; pp. 104–113.
11. Karbownik, P.; Russek, P.; Wiatr, K. Weak RSA Keys Discovery on GPGPU. *Int. J. Electron. Telecommun.* **2019**, *65*, 25–31. [CrossRef]
12. Overmars, A.; Venkatraman, S. Mathematical Attack of RSA by Extending the Sum of Squares of Primes to Factorize a Semi-Prime. *Math. Comput. Appl.* **2020**, *25*, 63. [CrossRef]
13. Ariffin, M.R.K.; Abubakar, S.I.; Yunos, F.; Asbullah, M.A. New Cryptanalytic Attack on RSA Modulus N=pq Using Small Prime Difference Method. *Cryptography* **2019**, *3*, 2. [CrossRef]
14. Yan, S.Y. Factoring Based Cryptography. In *Cyber Cryptography: Applicable Cryptography for Cyberspace Security*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 217–286.