Unprovable Security of RSA-OAEP in the Standard Model

Daniel R. L. Brown*

June 30, 2006

Abstract

Consider the provable security of RSA-OAEP when not instantiated with random oracles. Suppose a security reduction exists to show that finding a plaintext from a RSA-OAEP ciphertext (breaking the basic OW-CPA security) is as hard as the RSA problem.

- The reduction can be used in an adaptive chosen ciphertext text (IND-CCA2) attack against RSA-OAEP.
- The reduction cannot succeed in the random oracle model, so depends on how RSA-OAEP is instantiated.

Therefore, even the most basic security of RSA-OAEP without random oracles seems unprovable.

Key Words: RSA, OAEP, Provable Security, Public-key Encryption, IND-CCA2, OW-CPA, Impossibility Results

1 Introduction

Public-key encryption's most well-known provably secure scheme is RSA-OAEP (Rivest Shamir Adleman Optimal Asymmetric Encryption Padding). This scheme is proven [BR94, Sho01, FOPS01] to have semantic security against chosen ciphertext attacks, which is commonly abbreviated to IND-CCA2 security. This security property is widely considered the gold standard of security for public-key encryption. Earlier provably secure schemes were not as compact or as efficient as RSA-OAEP, which partially explains its popularity.

The main criticisms of the provable security of RSA-OAEP have been (a) that its security proof is only defined in the random oracle model, (b) there was long-standing misimpression about what security the original proof provided, which was ultimately resolved to the silver standard of IND-CCA1 so new proofs were devised for IND-CCA2, and (c) the reductions in the proofs are not tight, so for RSA-OAEP to achieve a provable security level of useful, larger public key sizes are needed. These criticism have been ordered in order of decreasing significance, which is coincidentally apparently also in order of increasing attention.

This paper examines the provable security of RSA-OAEP in the standard model, where random oracles are not allowed. Hypothetical security proofs for RSA-OAEP are then considered. Two types of proofs are considered. The first is a very simple type, perhaps the most natural one would consider, and the least one could expect. This first type is shown to imply a security weakness of RSA-OAEP. Indeed, the weakness is an IND-CCA2 attack, contradicting the existing security proofs. In fact, this show the IND-CCA2 security is difficult to prove in the standard model.

*Certicom Research

The second type of security proof considers a slight more ambitious type of proof. This proof aims to not depend on the specific instantiation of RSA-OAEP, but otherwise has the same goal as the first type of proof, which is the most basic security property of RSA-OAEP: the lowly "iron" standard, OW-CPA security, that an adversary cannot recover the plaintext by observation the ciphertext and public key only. It is shown that this proof is impossible. This suggests that any proof of the basic security must of the first type, which undermines the IND-CCA2 security.

2 Review of RSA-OAEP and OW-CPA Security

This section reviews the definition of RSA-OAEP and the definition of OW-CPA security of RSA-OAEP.

2.1 RSA-OAEP

We recall how RSA-OAEP works. It consists of a triple of algorithms (K, E, D).

- Algorithm K, key generation, takes a public exponent e and input seed value s (for randomization), and outputs an RSA modulus (public key) n and a RSA private exponent d. The public key n is such that $\gcd(e, \phi(n)) = 1$ and $n = pq$ where p and q are primes in a given fixed interval. A typical public exponent is $e = 2^{16} + 1$. The private exponent is such that $de \equiv 1 \bmod \phi(n)$. The detailed workings of K are not needed for the analysis here. We write $(n, d) = K(e, s)$.

- Algorithm E, public key encryption, takes an RSA public exponent e, and RSA public modulus n, a plaintext message m, and input value r (for randomization). The bit lengths of these of values is to be taken implicitly from below. The output is a ciphertext:

$$c = \left(\left((m\|z) \oplus G\left(r\right)\right) \| \left(r \oplus H\left((m\|z) \oplus G\left(r\right)\right)\right)\right)^e \bmod n \tag{1}$$

  where $\|$ indicates concatenation, z is fixed bit string (such as an all zero bit string), G and H are mask generation (hash) functions, and $\oplus$ indicates bit-wise exclusive-or. We write $c = E(e, n, m, r)$. The length of z should match the security level, so is usually at least 80 bits, which is necessary to ensure the plaintext awareness and other advanced security properties of OAEP. Note that G, H and z are parameters of the scheme, not inputs. The functions G and H are often modeled by random oracles in security proofs, but in practice are derived by combining counters and secure hash functions such as SHA-1.

- Algorithm D, private key decryption, takes an RSA public modulus n, an RSA private exponent d, and an alleged ciphertext c. It outputs a boolean value β and a message m, as follows. Let $t = c^d \bmod n$, and parse this as $t = u\|v$ into pieces of lengths matching G and H respectively. Let $r = H(u) \oplus v$. Let $w = G(r) \oplus u$, and parse this as $w = x\|y$, where y has the same bit length as the fixed bit string z. If $y \neq z$, then set output $\beta = 0$ (false) and set output m to some fixed error message. If $y = z$, then set output $\beta = 1$ (true) and output $m = x$, as the purported plaintext message. We write this as $(\beta, m) = D(n, d, c)$.

Certain minor details have been left out of the description above. For example, the length of the input to RSA encryption can be up to 8 bits shorter than that of the public modulus n, in which

case one effectively pads the base of the exponent with a leading zero octet. As one might expect, decryption undoes encryption, assuming the public key corresponds to the private key:

$$K(e, s) = (n, d) \Rightarrow D(n, d, E(e, n, m, r)) = (1, m). \tag{2}$$

2.2 OW-CPA and IND-CCA2 Security

Consider first the most basic type of adversary of an asymmetric (public-key) encryption scheme. This adversary takes a challenge ciphertext and outputs the corresponding plaintext. Indeed, RSA-OAEP was deliberately designed to resist much stronger adversaries, so presumably such adversaries ought not to exist. We formalize A as follows. In equation form, we have:

$$K(e, s) = (n, d) \Rightarrow A(e, n, E(e, n, m, r)) = m. \tag{3}$$

Where needed, we consider the success rate of A and the computation cost of A. Essentially, adversary A reproduces the decryption algorithm D, except that it has to use the public key instead of the private key. The action of A when fed invalid ciphertexts (for which D outputs $\beta = 0$), is irrelevant for determining the success of A.

The input to A is an RSA public exponent e, an RSA public modulus n consistent with e in that $\gcd(e, \phi(n)) = 1$, and a valid ciphertext c, meaning a ciphertext that would be obtained by randomly selecting a message m from some message space, and random value of r of the designate length, and computing $c = E(e, n, m, r)$. For the purposes of this definition, we use a message space large enough so that if A guesses a random message in the space, then A has negligible chance of guessing a given message m. We may also require that the space be efficiently sampleable. Note that A, as described, is a probabilistic algorithm, but we can also model it as a deterministic function if we give it an additional input, a randomizing value.

In IND-CCA2 security, the adversary A gets the public key, and then chooses two messages m_0 and m_1 which are submitted to an encryption oracle, who picks random $b \in \{0, 1\}$ and encrypts message m_b as ciphertext c_b. The adversary is given c_b and has the goal of finding b. The adversary gets access to a decryption oracle that will decrypt any valid ciphertext except c_b.

2.3 Security Reductions for the OW-CPA Security of RSA-OAEP

Consider a certain kind of reduction R that uses an OW-CPA adversary A to RSA-OAEP as described above. The reduction R is an algorithm that takes input of a public exponent e, a RSA public modulus n consistent with e, and random integer $y \in [0, n-1]$. Algorithm R then attempts to output x such that

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

which is known as solving the RSA problem. Algorithm R gets to call algorithm A as a subroutine. When calling A, reduction R must supply e and n input values for A identical the values given to R, but it may choose any value for the ciphertext c. Assuming that A is a successful adversary and the chosen ciphertext is valid, then A will decrypt and give the answer to R. If the ciphertext is invalid, then the output of a successful adversary A need not satisfy any conditions.

We assume that reduction R is successful whenever A is (or more generally has a success rate at least proportional to that of A). We make no restrictions on what A can do. In particular, when G and H are instantiated with specific hash functions, then A is permitted to process their descriptions and make computations upon them. In other words, A is not required to treat G and

H as random oracles. A specific adversary A may work only for the single pair G and H are not for any other pair of hash functions. Even with such an adversary A, reduction R must be able to use A to the solve the RSA problem.

We will call this type of reduction a *simple reduction*. We note that a simple reduction is arguably too simple, since it does not seem to depend in any way on the security of G and H. Therefore, we will also consider some other kinds of reduction.

2.4 Weak Reductions: Dependence on the Instantiation

One way to handle G and H in the reduction is to modify the goal of the reduction: solve either the RSA problem or defeat the security of G or H, under some arbitrary definition. This better reflects the intuition that the security of RSA-OAEP depends to some extent on the security of G and H.

We will call this type of reduction, a *weak reduction*. Intrinsic to the idea of weak reduction, is the separation of the RSA problem from the security of G and H. One may define even weaker reductions that do not make such a separation or that substitute a potentially easier problem for the RSA problem (whose hardness is thus a stronger assumption). Such weaker reductions shall not be considered formally in this paper.

2.5 Strong Reductions: Succeeding in the Random Oracle Model

We shall also consider a powerful type of reduction R. This type of reduction is not specific to G and H. To formalize this precisely, we restrict the access of R to G and H to be random oracle access. The success rate of R is defined over random choices of G and H. Provided that for a random choice of G and H, we can supply a successful adversary A, then R should be able to solve the RSA problem, with a reasonable chance of success. One can easily test any given reduction R algorithm by simulating A using an RSA private key and simulating random oracles G and H for R.

We will call this type of reduction a *strong reduction*. A strong reduction is desirable for three reasons. The first is its power. It suggests that if G and H appear random enough (to the reduction), then RSA-OAEP is secure even if the adversary is not limited to random oracle access to G and H. The second reason is that the alternative seems implausible: if the reduction does not treat G and H like random oracles, then it would seem to be depend on properties of G and H that do not hold for random functions. This defies the existing intuition that G and H are most secure when instantiated as random functions. A third reason is to avoid our metareduction against simple and weak reductions.

3 A Metareduction Against Simple and Weak Reductions

We now describe a metareduction M that takes a simple reduction R for the OW-CPA security of RSA-OAEP and uses it to break the IND-CCA2 security of RSA-OAEP. The idea is simple: M acts as an IND-CCA2 adversary, calling R to solve RSA problem upon a randomized version of its challenge ciphertext, answering the queries of R using its decryption oracle. This also gives rise to a OW-CCA2 attack, not just an IND-CCA2 attack.

Figure 1 illustrates the details of M. First M receives a challenge public key (n, e) generated by K. Then M selects two distinct messages m_0 and m_1, as required for an IND-CCA2. It does not matter what these values are for M to succeed, so the selection may be arbitrary. Next these two

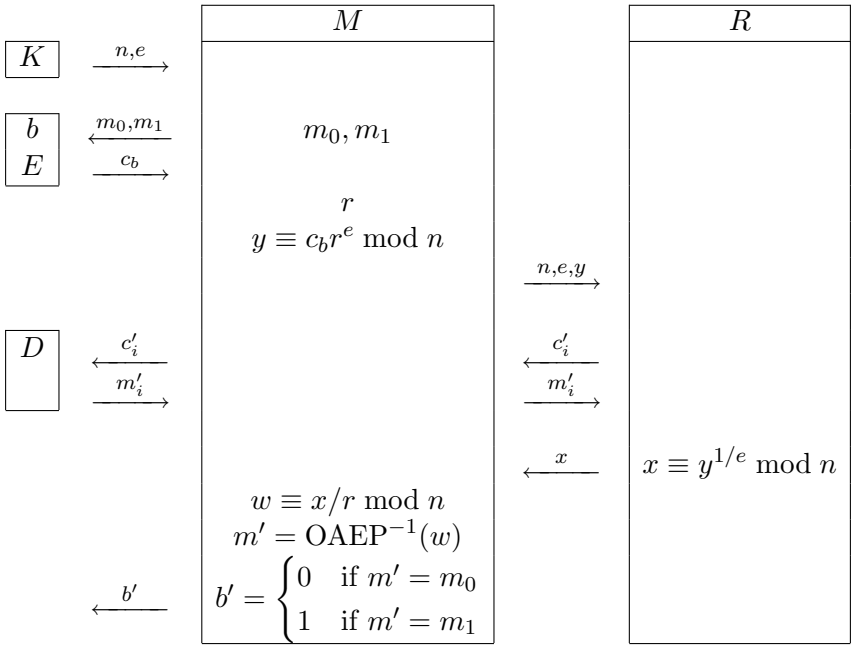

Figure 1: Converting OW-CPA Reduction to IND-CCA2 Adversary

messages are sent to an encryption oracle that randomly chooses a value $b \in \{0, 1\}$, and the applies E to m_b to obtain a ciphertext c_b. Then M receives c_b and its goal is to find b. In the OW-CCA2 setting, M would instead receive the encryption of a message selected at random from some large message space. Now M picks a random value r, and computes $y \equiv c_b r^e \bmod n$, which effectively produces a random RSA challenge value to be sent to reduction R. Reduction R requires access to an OW-CPA adversary, to which it sends ciphertext queries c'_i, and from which it expects decryption m'_i of the ciphertext queried. To answer the ciphertext queries from R, the metareduction passes them on to the decryption oracle, which it has by virtue of playing the role of an IND-CCA2 (or OW-CCA2) adversary. We see that R has negligible chance of choosing $c'_i = c_b$, so therefore the decryption oracle is allowed to decrypt the ciphertexts as m'_i. Now, the metareduction returns these back to the reduction R. Iterating as many times as needed, eventually all the queries of R are answered correctly, and R solves the RSA problem with a value x such that $x^e \equiv y \bmod n$. Metareduction computes $w = x/r \bmod n$, which satisfies $w^e = c_b \bmod n$. This means that w is the OAEP padding of some message, and that the OAEP padding can be removed to recover a message $m' = \text{OAEP}^{-1}(w)$. We have $m' \in \{m_0, m_1\}$ and now the metareduction can recover b.

Suppose the reduction is weak, as defined earlier, so that it is possible that R does not solve the RSA problem but instead breaks the security of G or H according to some definition. Then it seems that M, as described above, succeeds in breaking the IND-CCA2 security of RSA-OAEP or succeeds in breaking the security of G or H. Assuming that RSA-OAEP is IND-CCA2 secure, then M can break the security of G or H, in which case the security reduction R is not useful, because it rests on a broken security property of G or H. If the G and H are secure, in whatever defined sense, then it is safe to say that M cannot break the security of G or H, and therefore it can defeat the IND-CCA2 security of RSA-OAEP.

Metareduction M applies to other public-key encryption schemes and related reductions. The

reduction needs to show the OW-CPA security of the encryption scheme is related to a hard problem with certain properties. The hard problem must be a necessary for the OW-CPA security in the sense that be able to solve the problem implies a OW-CPA attack, and the problem has to be blindable: one can randomize an instance of the problem, solve the randomized instance, then use the solution to solve the original problem. In particular, the metareduction M works against RSA-KEM.

The metareduction M also implies that a reduction R' in the standard model that shows breaking the IND-CCA2 security of RSA-OAEP is as hard as the RSA problem cannot exist, unless the RSA is insecure. If such a reduction R' existed, one can translate it into a reduction R showing that the OW-CPA security of RSA-OAEP is hard as the problem RSA problem. In the translation, reduction R simulates an IND-CCA2 adversary to R', without making any chosen ciphertext (decryption oracle) queries. Reduction R selects any two messages for R for the encryption oracle query that R' is expecting. Reduction R calls its decryption oracle, the adversary it is expecting to use as subroutine, to decrypt the challenge ciphertext generated by R' as a response to the encryption oracle query. The decryption enables R to determine which message was encrypted. Therefore R will appear to be a successful IND-CCA2 adversary to R', and R' will solve the given instance of the RSA problem on behalf of R. Given this reduction R, now metareduction M can be used to give an attack on the IND-CCA2 security of RSA-OAEP, and then reduction R' on this attack can be used to solve the RSA problem, so RSA would be insecure. It is more plausible to conclude that RSA-OAEP cannot be proven IND-CCA2 secure in the standard model under only the assumption that RSA problem is hard (and some reasonable assumption about G and H). This does not mean that RSA-OAEP is IND-CCA2 insecure, or that a security proof is impossible. Perhaps a reduction involving a modified variant of the RSA problem should be sought.

4 A Metareduction Against Strong Reductions

We define a metareduction M that uses a strong reduction, as defined earlier, to solve the RSA problem. Therefore if the RSA problem is hard, no strong reduction R exists. If a strong reduction exists, then RSA problem is easy, and therefore the security proof using the strong reduction provides no assurance for RSA-OAEP.

The metareduction M is given a random challenge instance of the RSA problem, and to solve this challenge, M simply asks the reduction R to solve the challenge. The reduction R has the same objective as the metareduction M, so this is fine. The reduction R, however, expects to talk to an adversary A to the RSA-OAEP scheme, specifically an adversary A that can take valid RSA-OAEP ciphertext and decrypt it. The metareduction must be able to play the role of the adversary of A, or else, R is not guaranteed to solve the RSA problem challenge.

In order to be able to decrypt the ciphertexts that R expects A to decrypt, the metareduction employs the random oracle model. Metareduction M runs R with a random oracle, randomly selecting and recording the outputs of the outputs of the random oracles. When R generates a claimed ciphertext c, there is negligible chance that it is valid unless R has already queried the corresponding inputs associated with the plaintext and OAEP randomizer to the random oracle.

When M sees a ciphertext query from R, it can search the random oracle queries, re-encrypting to compare to c. If c does not match any pair of random oracle queries, then M reports the ciphertext to be invalid, and otherwise M provides the decryption, which is visible from the random oracles queries made by R. Now M is able to answer the RSA-OAEP decryption queries made by R. Note that this is the same principle of plaintext awareness used in Bellare and Rogaway's

[BR94] original proof of security for RSA-OAEP. Reduction R thinks it speaking to an adversary A, and therefore R solves the RSA challenge on behalf of the metareduction M, as required.

A formal description of the metareduction M follows. Its input is a public exponent e, a corresponding RSA modulus n, and a random integer $y \in [0, n-1]$. This is the same input as for R. More precisely, e is a fixed value, and n is obtained from $(n, d) = K(e, s)$ for some random choice of s. Next, metareduction M runs reduction R on inputs e and n. Algorithm R sends queries to subroutine A, G and H. Metareduction M answers these queries as follows:

- If a G-query input is new, then M chooses a random output value of the correct length, and responds to R with this value. For old queries, M responds with the previous output.
- If an H-query input is new, then M chooses a random output of the correct length, and response to R with this value. For old queries, M responds with the previous output.
- If an A-query input is new, then M goes through all past G and H inputs. For each pair, say r to G and s to H, compute
$$c = (s \| (r \oplus H(s)))^e \bmod n \tag{5}$$
If c equals the ciphertext input to A, then compute $t = s \oplus G(r)$ and parse this is $t = x \| y$, where y is the length of z. If $y = z$, then answer the A-query with message $m = x$. Otherwise, that is, if $y \neq z$, or if no value of c matches the A-query input, then answer the A-query arbitrarily, such as with a random message or an error message.

After having all its queries answered, reduction R will either fail or succeed, solving the RSA problem for instance (e, n, y) giving a solution x. If reduction R succeeds, then metareduction M succeeds by outputting the solution x to its challenge RSA problem.

We need to prove R cannot distinguish the answers that M provides from true random oracles for G and H and a true adversary A. The metareduction responses to hash function queries to G and H are chosen purely at random, so therefore are statistically indistinguishable from random oracles.

The queries to the adversary A are more interesting. The ciphertext queries of R can be classified in two types. The first type is that one for which the metareduction can find r and s and compute the plaintext. It is easy to see that M responds to correctly to these A-queries, and that R cannot use these to detect that A is being simulated. The second type of ciphertext query to the A-oracle is one in which the corresponding the r and s values (that exist) were not submitted to the G-oracle and H-oracle, respectively.

We argue that the second type of ciphertext query has negligible probability of being a valid ciphertext, and therefore, reduction R cannot feasibly expect that the second type of ciphertext to be valid, and therefore cannot feasibly expect to gain anything from the responses. More to the point, there is negligible probability that the second type of query will be indistinguishable from the case when G and H are random oracles and A is true adversary to RSA-OAEP. This is because if R were to somehow generate a valid ciphertext of the second type of ciphertext it would have to guess in advance correctly the outputs of the functions G and H which it has not yet seen, and the probability of guessing them correctly is negligible.

5 Insecure Instantiations

In this section are considered some contrived insecure instantiations of G and H in RSA-OAEP and their impact on its OW-CPA security. They serve to demonstrate that the particular instantiation

of RSA-OAEP has some relevance, and that a security proof in the standard model should consider G and H.

5.1 An Impractical Instantiation

The following is an impractical instantiation, because H is not efficiently computable. Set up the parameters of RSA-OAEP so that r and z have the same bit length, and choose the length of these to very close to the length of m. Choose G to be a zero constant value padded with the identity function. Therefore $(m\|z) \oplus G(r) = m\|(z \oplus r)$. The function H is defined implicitly by

$$H(m\|(r \oplus z)) = h \tag{6}$$

where h is a bit string such that

$$(m\|(r \oplus z)\|(r \oplus h))^e \equiv m\|w \bmod n \tag{7}$$

for some bit string w, if such an h exists. We call this a good case. If an h with this property does not exist, then the output can be arbitrary, and this is the bad case.

We argue heuristically, that the good case for H occurs fairly frequently. Modeling the function $x \mapsto x^e \bmod n$ as random, we expect that for random choice of h, the probability of (7) is one over the number of choices for m. But the number of choices for h is the same as the number of choices for m, and thus we expect that about one choice of h will cause (7) to hold.

In the good case, the ciphertext generated upon encryption of m will contain the message m, padded with some nonsense string w. This is clearly insecure in the OW-CPA sense, since with a good probability the adversary can just read the plaintext immediately off the ciphertext.

5.2 A Practical Instantiation

Let G be a constant zero function, let $e = 3$, let z have length more than two-thirds of the ciphertext length. A ciphertext then has the form $c \equiv (m\|z\|s)^3 \bmod n$, for some unknown string s. It seems plausible that algorithms similar to Coppersmith's [Cop96] can be used to solve for m given c. These algorithms invert the RSA function when given sufficient information about the input to the RSA function, which in this case is represented by the fixed value z.

6 Conclusion

Metareductions show that it is difficult to prove the OW-CPA security of RSA-OAEP in the standard model.

Acknowledgments

I thank Alfred Menezes for encouragement and suggestions.

References

[BR94] Mihir Bellare and Phillip Rogaway, *Optimal asymmetric encryption*, Advances in Cryptology — EUROCRYPT '94 (Alfredo De Santis, ed.), LNCS, no. 950, IACR, Springer, May 1994, pp. 92–111.

[BV98] Dan Boneh and Ramarathnam Venkatesan, *Breaking RSA may be easier than factoring*, Advances in Cryptology — EUROCRYPT '98 (Kaisa Nyberg, ed.), LNCS, no. 1403, IACR, Springer, May 1998, `http://crypto.stanford.edu/~dabo/abstracts/no_rsa_red.html`, pp. 59–71.

[Cop96] Don Coppersmith, *Finding a small root of a univariate modular equation*, Advances in Cryptology — EUROCRYPT '96 (Ueli Maurer, ed.), LNCS, no. 1070, IACR, Springer, May 1996, pp. 155–165.

[FOPS01] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern, *RSA-OAEP is secure under the RSA assumption*, in Kilian [Kil01], See `http://eprint.iacr.org/2000/061`, pp. 260–274.

[Kil01] Joe Kilian (ed.), *Advances in cryptology — CRYPTO 2001*, LNCS, no. 2139, IACR, Springer, August 2001.

[PV05] P. Paillier and D. Vergnaud, *Discrete-log-based signatures may not be equivalent to discrete log*, Advances in Cryptology — ASIACRYPT 2005 (Bimal Roy, ed.), LNCS, no. 3788, IACR, Springer, December 2005, pp. 1–20.

[Sho01] Victor Shoup, *OAEP reconsidered*, in Kilian [Kil01], See `http://eprint.iacr.org/2000/060`, pp. 239–259.

A Similar Work on Metareductions and OAEP

Metareductions are not new. Boneh and Venkatesan [BV98] give a metareduction that essentially shows there is no reduction proving the equivalence of the low exponent RSA problem and integer factorization. More precisely, they show there is no algebraic reduction, which is somewhat analogous to our result that there is no strong reduction. Paillier and Vergnaud [PV05], who won the best paper award at Asiacrypt 2005, give some similar results in the discrete log setting.

Shoup gives [Sho01] an argument that the design of OAEP cannot be proven secure when RSA is replaced by an arbitrary trapdoor one-way function. He hypothesizes a pathological trapdoor one-way function for which OAEP becomes insecure. This is more akin to the insecure instantiations of RSA-OAEP proposed in this paper than to a metareduction.

The phenomenon of security proofs leading to attacks is not new either. Rabin's digital signatures and public key encryption had reductions showing that their the basic security against passive adversaries was equivalent to the integer factorization problem. It was soon discovered, though, that these reduction algorithms also led to an active attacks. Countermeasures have since been discovered to thwart the active attacks.