Submission to Third NIST Workshop on Block Cipher modes of operation.

Upgrading AEAD Privacy: The AE2 goal

Mihir Bellare

Department of Computer Science and Engineering
University of California San Diego
mbellare@ucsd.edu

June 2023

This talk will argue that the privacy attributes of current AEAD schemes are weaker than desirable in some important contexts. It suggests that we standardize schemes meeting a stronger definition called AE2. The content here is based on [2].

NBE1 and AE1-security. We refer to the syntax of the current form of symmetric encryption [21, 23] as NBE1. An NBE1 scheme $\mathsf{SE1}$ specifies a *deterministic* encryption algorithm $\mathsf{SE1.Enc}$ that takes the key K, a nonce N, message M and a header (also called associated data) H to return what we call a core ciphertext C_1. Deterministic decryption algorithm $\mathsf{SE1.Dec}$ takes K, N, C_1, H to return either a message or $\perp$.

Security asks for privacy of M and integrity of both M and H *as long as nonces are unique*, meaning not re-used. Rogaway's formalization [21] asks that an adversary given oracles for encryption (taking nonce, message and header) and decryption (taking nonce, core ciphertext and header) be unable to distinguish between the case where they perform their prescribed tasks under a hidden key, and the case where the former returns random strings and the latter returns $\perp$, as long as the adversary does not repeat a nonce across its encryption queries. We will refer to this as basic AE1-security.

NBE1 providing basic AE1-security has been the goal of recent schemes, standards and proposed standards, as witnessed by GCM [17, 8] (used in TLS), OCB [25, 22, 15], CAESAR candidates [4] and RFC 5116 [16]. The security of NBE1, which we revisit, is thus of some applied interest.

The gap. Our concern is a gap between theory and usage that can result in privacy vulnerabilities in the latter. Recall that the decryption algorithm $\mathsf{SE1.Dec}$, to be run by the receiver, takes as input not just the key K, core ciphertext C_1 and header H, but *also the nonce* N. The theory says that how the receiver gets the nonce is "outside of the model" [21] or that it is assumed to be communicated "out-of-band" [23]. Usage cannot so dismiss it, and must find a way to convey the nonce to the receiver. The prevailing understanding, reflected in the following quote from RBBK [25], is that this is a simple matter— if the receiver does not already have the nonce N, just send it in the clear along with the core ciphertext C_1:

The nonce N is needed both to encrypt and to decrypt. Typically it would be communicated, in the clear, along with the (core) ciphertext.

RFC 5116 is a draft standard for an interface for authenticated encryption [16]. It also considers it fine to send the nonce in the clear:

> ... there is no need to coordinate the details of the nonce format between the encrypter and the decrypter, as long *the entire nonce is sent* or stored with the ciphertext and is thus available to the decrypter ... the nonce MAY be stored *or transported* with the ciphertext ...

To repeat and summarize, the literature and proposed standards suggest transmitting what we call the "full" ciphertext, consisting of the nonce and the core ciphertext. Yet, as we now explain, this can be wrong.

Nonces can compromise privacy. We point out that communicating a nonce in the clear with the ciphertext can damage, or even destroy, message privacy. One simple example is a nonce $N = F(M)$ that is a hash —under some public, collision-resistant hash function F— of a low-entropy message M, meaning one, like a password, which the attacker knows is likely to fall in some small set or dictionary D. Given a (full) ciphertext $C_2 = (N, C_1)$ consisting of the core ciphertext $C_1 = \mathsf{SE1.Enc}(K, N, M, H)$ together with the nonce $N = F(M)$, the attacker can recover M via "For $M' \in D$ do: If $F(M') = N$ then return M'." To take a more extreme case, consider that the nonce is some part of the message, or even the entire message, in which case the full ciphertext clearly reveals information about the message.

The concern that (adversary-visible) nonces compromise privacy, once identified, goes much further. Nonces are effectively meta-data. Even recommended and innocuous-seeming choices like counters, device identities, disk-sector numbers or packet headers reveal information about the system and identity of the sender. For example, the claim that basic-AE1-secure NBE1 provides anonymity —according to [24, Slide 19/40], this is a dividend of the requirement that core ciphertexts be indistinguishable from random strings— is moot when the nonce includes sender identity. Yet the latter is not only possible but explicitly recommended in RFC 5116 [16], which says: "When there are multiple devices performing encryption ... use a nonce format that contains a field that is distinct for each one of the devices." As another concrete example, counters are *not* a good choice of nonce from a user privacy perspective, as pointed out by Bernstein [5] and the ECRYPT-CSA *Challenges in Authenticated Encryption* report [1].

The above issues apply to all NBE1 schemes and do not contradict their (often, proven) AE1-security. They are not excluded by the unique nonce requirement or by asking for misuse resistance [26], arising in particular for the encryption of a single message with a single corresponding nonce.

A natural critique is that the privacy losses we have illustrated occur only for "pathological" choices of nonces, and choices made in practice, such as random numbers or counters, are "fine." This fails, first, to recognize the definitional gap that allows the "pathological" choices. With regard to usage, part of the selling point of NBE1 was exactly that *any* (non-repeating, unique) nonce is fine, and neither existing formalisms [21] nor existing standards [16] preclude nonce choices of the "pathological" type. Also, application designers and users cannot, and should not, carry the burden of deciding which nonces are "pathological" and which are "fine," a decision that may not be easy. (And as discussed above, for example, counters may *not* be fine.) Finally, Section **??** indicates that poor choices can in fact arise in practice.

Perspective. Our perspective is that the above issues reflect a gap between the NBE1 formalism and the privacy provided by NBE1 in usage. We next discuss how to bridge it, following [2]. This involves a modified syntax for nonce-based encryption, called NBE2, in which decryption does not get the nonce, a corresponding framework of security definitions called AE2 that guarantee nonce privacy in addition to authenticity and message privacy, and simple ways to turn NBE1 AE1-secure schemes into NBE2 AE2-secure schemes.

AE2-secure NBE2 obviates application designers and users from the need to worry about privacy implications of their nonce choices, simplifying design and usage. With AE2-secure NBE2, one can use any nonce, even a message-dependent one such as a hash of the message, without compromising privacy of the message. And the nonces themselves are hidden just as well as messages, so user-identifying information in nonces doesn't actually identify users.

THE NBE2 SYNTAX. In an NBE2 scheme $\mathsf{SE2}$, the inputs to the deterministic encryption algorithm $\mathsf{SE2.Enc}$ continue to be key K, nonce N, message M and header H, the output C_2 now called a ciphertext rather than a core ciphertext. The deterministic decryption algorithm $\mathsf{SE2.Dec}$ *no longer gets a nonce*, taking just key K, ciphertext C_2 and header H to return either a message M or $\perp$.

Just as an interface, NBE2 already benefits application designers and users, absolving them of the burden they had, under NBE1, of figuring out and architecting a way to communicate the nonce from sender to receiver. The NBE2 receiver, in fact, is nonce-oblivious, not needing to care, or even know, that something called a nonce was used by the sender. By reducing choice (how to communicate the nonce), NBE2 reduces error and misuse.

THE AE2-SECURITY FRAMEWORK. The AE2 game gives the adversary an encryption oracle $\textsc{Enc}$ (taking nonce N, message M and header H to return a ciphertext C_2) and decryption oracle $\textsc{Dec}$ (as per the NBE2 syntax, taking ciphertext C_2 and header H but no nonce, to return either a message M or $\perp$). When the challenge bit is $b = 1$, these oracles reply as per the encryption algorithm $\mathsf{SE2.Enc}$ and decryption algorithm $\mathsf{SE2.Dec}$ of the scheme, respectively, using a key chosen by the game. When the challenge bit is $b = 0$, oracle $\textsc{Enc}$ returns a ciphertext that is drawn at random from a space $\mathsf{SE2.CS}(|N|, |M|, |H|)$ that is prescribed by the scheme $\mathsf{SE2}$ and that depends only on the lengths of the nonce, message and header, which guarantees privacy of both the nonce and message. (This space may be, but unlike for AE1 need not be, the set of all strings of some length, because NBE2 ciphertexts, unlike NBE1 core ciphertexts, may have some structure.) In the $b = 0$ case, decryption oracle $\textsc{Dec}$ returns $\perp$ on any non-trivial query. The adversary eventually outputs a guess b' as to the value of b, and its advantage is $2\Pr[b = b'] - 1$.

We say that $\mathsf{SE2}$ is AE2[$\mathcal{A}$]-secure if practical adversaries in the class $\mathcal{A}$ have low advantage. Let $\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}$ be the class of $\underline{\text{u}}$nique-$\underline{\text{n}}$once adversaries, meaning ones that do not reuse a nonce across their $\textsc{Enc}$ queries. We refer to AE2[$\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}$]-security as basic AE2-security. As the nonce-hiding analogue of basic AE1-security, it will be our first and foremost target.

BUILDING AE2 SCHEMES. Towards AE2 schemes for standardization, we have two options. One is to build such schemes from scratch. BNT [2] give an easier path. They provide simple, cheap transforms that turn a given AE1-secure NBE1 schemes into an AE2-secure NBE2 scheme. We currently have a portfolio of efficient AE1-secure NBE1 schemes supported by proofs of security with good concrete bounds [25, 17, 4, 15, 13, 28, 19, 10, 20, 9, 6, 12]. Standards can leverage this, using transforms from [2] to obtain options for AE2-secure NBE2 schemes. We now overview the transforms.

Since NBE2 schemes effectively take care of nonce communication, we expect ciphertext length to grow by at least $\mathsf{SE1.nl}$, the nonce length of the base NBE1 scheme. The *ciphertext overhead* is defined as the difference between the ciphertext length and the sum of plaintext length and $\mathsf{SE1.nl}$. All the transforms from [2] have zero ciphertext overhead. One challenge in achieving this is that nonce lengths like $\mathsf{SE1.nl} = 96$ are widely-used but short of the block length 128 of many blockciphers, precluding inclusion of an extra blockcipher output in the ciphertext. With regard to

NBE2 scheme	**AE2-security provided**	
	Basic	Advanced
$\mathbf{HN1}[\mathsf{SE1},\mathsf{F}]$	Yes	Yes
$\mathbf{HN2}[\mathsf{SE1},\ell,\mathsf{E},\mathsf{Spl}]$	Yes	Yes if $\ell \geq 128$
$\mathbf{HN3}[\mathsf{SE1},\mathsf{F}]$	Yes	No
$\mathbf{HN4}[\mathsf{SE1},\ell,\mathsf{F}]$		Yes
$\mathbf{HN5}[\mathsf{TE},\ell,\ell_z]$		Yes

Figure 1: Security attributes of the NBE2 schemes defined by the Hide-Nonce (HN) transforms. In the table $\mathsf{SE1}$ denotes an NBE1 scheme, F a PRF, E a block cipher, and TE a variable-length tweakable block cipher. Spl is a splitting function, and ℓ, ℓ_z are non-negative integer parameters. A blank entry in the Basic column means the transform is not for that purpose. Note that **HN1**'s advanced security only holds when ciphertexts have sufficiently large (e.g. 128 bits) minimum length, and **HN2**'s depends on the length of the stolen ciphertext.

---

computational overhead, the challenge is that it should be constant, meaning independent of the lengths of the message and header for encryption, and of the ciphertext and header for decryption. All the transforms also have constant computational overhead.

The following discussion first considers achieving basic security and then advanced security, meaning the misuse resistance goal of [26]. Security attributes of the corresponding "Hide-Nonce (HN)" transforms are summarized in Figure 1.

Basic HN transforms. All the following transforms from [2] turn a basic-AE1-secure NBE1 scheme $\mathsf{SE1}$ into a basic-AE2-secure NBE2 scheme $\mathsf{SE2}$. (Recall basic means nonces are unique, never reused across encryption queries.)

Having first produced a core ciphertext C_1 under $\mathsf{SE1}$, the idea of scheme $\mathsf{SE2} = \mathbf{HN1}[\mathsf{SE1},\mathsf{F}]$ is to use C_1 itself as a nonce to encrypt the actual nonce in counter mode under PRF F. A drawback is that this requires the minimal core-ciphertext length $\mathsf{SE1.mccl}$ to be non-trivial, like at least 128, which is not true for all $\mathsf{SE1}$. Scheme $\mathsf{SE2} = \mathbf{HN2}[\mathsf{SE1},\ell,\mathsf{E},\mathsf{Spl}]$ turns to the perhaps more obvious idea of enciphering the nonce with a PRF-secure blockcipher E. The difficulty is the typicality of 96-bit nonces and 128-bit blockciphers, under which naïve enciphering would add a 32-bit ciphertext overhead, which is resolved by ciphertext stealing, ℓ representing the number of stolen bits (32 in our example) and Spl an ability to choose how the splitting is done. Scheme $\mathsf{SE2} = \mathbf{HN3}[\mathsf{SE1},\mathsf{F}]$ uses the result of PRF F on the actual nonce as a derived nonce under which to run $\mathsf{SE1}$. This is similar to SIV [26, 19]; the difference is to achieve AE2 rather than AE1 and to apply the PRF only to the nonce (rather than nonce, message and header) to have constant computational overhead.

Advanced HN transforms. Unique nonces are easier to mandate in theory than assure in practice, where nonces may repeat due to errors, system resets, or replication. In that case (returning here to NBE1), not only does basic AE1-security give no security guarantees, but also damaging attacks are possible for schemes including CCM and GCM [14, 27]. Rogaway and Shrimpton's misuse resistant NBE1, which we refer to as advanced-AE1-secure NBE1, minimizes the damage from reused nonces, retaining AE1-security as long as no nonce, message, header triple is re-encrypted [26]. This still being for the NBE1 syntax, however, the concerns with adversary-visible nonces compromising message and user privacy are unchanged. We seek the NBE2 analogue, correspondingly defining and achieving advanced-AE2-secure NBE2 to provide protection against reused nonces while also hiding them.

With the above framework, the definition is easy, calling for no new games; the goal is simply AE2[$\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}nmh}}$]-security where $\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}nmh}}$ is the class of $\underline{\text{u}}$nique-$\underline{\text{n}}$once, $\underline{\text{m}}$essage, $\underline{\text{h}}$eader adversaries, meaning ones that do not repeat a query to their ENC oracle. The presence of well-analyzed advanced-AE1-secure NBE1 schemes [26, 11, 10, 9, 6] again motivates transforms rather than from-scratch designs.

We start by revisiting the above basic-security preserving transforms, asking whether they also preserve advanced security, meaning, if the starting NBE1 scheme is advanced-AE1-secure, is the transformed NBE2 scheme advanced-AE2-secure? For **HN1**, the answer is YES. It is YES also for **HN2** as long as the amount ℓ of stolen ciphertext is large enough. (In practical terms, at least 128.) For **HN3**, the answer is NO.

That **HN1** and **HN2** have these properties is good, but there are limitations. (Namely **HN1** puts a lower bound on $\mathsf{SE1.mccl}$ that is not always met, and setting $\ell = 128$ in **HN2** with typical 96-bit nonces will call for a 224-bit blockcipher.) Better methods are **HN4** and **HN5** [2]. Scheme $\mathsf{SE2} = \mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ uses the result of PRF F on the actual nonce, message and header as a derived nonce for $\mathsf{SE1}$. The difference with SIV [26, 19] is that what is encrypted under $\mathsf{SE1}$ includes the actual nonce in order to hide it. The computational overhead stays constant because $\mathsf{SE1}$ need provide only privacy, which it can do in one pass. Scheme $\mathsf{SE2} = \mathbf{HN5}[\mathsf{TE}, \ell, \ell_z]$ is different, using the encode-then-encipher paradigm [3] to set the ciphertext to an enciphering, under an arbitrary-input-length, tweakable cipher TE, of the nonce, message and ℓ_t-bits of redundancy, with the header as tweak. Instantiating TE via the very fast AEZ tweakable block cipher [11] yields correspondingly fast, advanced-AE2-secure NBE2.

DEDICATED TRANSFORM FOR GCM. While the above generic transforms are already able, with low overhead, to immunize GCM [17, 8] —by this we mean turn this basic-AE1-secure NBE1 scheme into a basic-AE2-secure NBE2 scheme— there is a dedicated transform —one that exploits the structure of GCM— that does even better. The goal is not just even lower cost overhead, but minimization of software changes. It is shown in [2] that simply pre-pending a block of 0s, of length equal to the nonce length, to the message, and then GCM-encrypting, provides basic-AE2-security. This means no new key materiel needs to be added, and existing encryption software can be used in a blackbox way. Ciphertext overhead remains zero. Decryption software does however need a change.

DISCUSSION. In a 2013 mailing list message, Bernstein [5] argues that the security definitions for authenticated encryption fail to fully capture practical requirements, giving sequence privacy leakage via sequence-number nonces as an explicit example. AE2-secure NBE2 addresses these concerns. Bernstein also proposed a solution that can be seen as a specific instantiation of our **HN2** transformation.

The CAESAR competition's call for authenticated encryption schemes describes a syntax where encryption receives, in place of a nonce, a public message number (PMN) and a secret message number (SMN), decryption taking only the former [7]. The formalization of Namprempre, Rogaway and Shrimpton (NRS) [18] dubs this "AE5." In this light, an NBE1 scheme is a AE5 scheme without a SMN and an NBE2 scheme is an AE5 scheme without a PMN.

References

[1] J. Aumasson, S. Babbage, D. Bernstein, C. Cid, J. Daemen, O. Dunkelman, K. Gaj, S. Gueron, P. Junod, A. Langley, D. McGrew, K. Paterson, B. Preneel, C. Rechberger,

V. Rijmen, M. Robshaw, P. Sarkar, P. Schaumont, A. Shamir, and I. Verbauwhede. CHAE: Challenges in authenticated encryption. ECRYPT-CSA D1.1, Revision 1.05, March 2017. `https://chae.cr.yp.to/whitepaper.html`. 2

[2] M. Bellare, R. Ng, and B. Tackmann. Nonces are noticed: AEAD revisited. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 235–265. Springer, Heidelberg, Aug. 2019. 1, 2, 3, 4, 5

[3] M. Bellare and P. Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 317–330. Springer, Heidelberg, Dec. 2000. 5

[4] D. Bernstein. CAESAR call for submissions, final (2014.01.27), 2014. 1, 3

[5] D. J. Bernstein. Re: secret message numbers. Message in Google group on cryptographic competitions, October 2013. `https://groups.google.com/d/msg/crypto-competitions/n5ECGwYr6Vk/bsEfPWqSAU4J`. 2, 5

[6] P. Bose, V. T. Hoang, and S. Tessaro. Revisiting AES-GCM-SIV: Multi-user security, faster key derivation, and better bounds. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 468–499. Springer, Heidelberg, Apr. / May 2018. 3, 5

[7] CAESAR Committee. Cryptographic competitions: Caesar call for submissions, final (2014.01.27). `https://competitions.cr.yp.to/caesar-call.html`. Accessed: 2018-07-23. 5

[8] M. Dworkin. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D, November 2007. 1, 5

[9] S. Gueron, A. Langley, and Y. Lindell. AES-GCM-SIV: Specification and analysis. Cryptology ePrint Archive, Report 2017/168, 2017. `https://eprint.iacr.org/2017/168`. 3, 5

[10] S. Gueron and Y. Lindell. GCM-SIV: Full nonce misuse-resistant authenticated encryption at under one cycle per byte. In I. Ray, N. Li, and C. Kruegel, editors, *ACM CCS 2015*, pages 109–119. ACM Press, Oct. 2015. 3, 5

[11] V. T. Hoang, T. Krovetz, and P. Rogaway. Robust authenticated-encryption AEZ and the problem that it solves. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 15–44. Springer, Heidelberg, Apr. 2015. 5

[12] V. T. Hoang, S. Tessaro, and A. Thiruvengadam. The multi-user security of GCM, revisited: Tight bounds for nonce randomization. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 1429–1440. ACM Press, Oct. 2018. 3

[13] T. Iwata, K. Ohashi, and K. Minematsu. Breaking and repairing GCM security proofs. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 31–49. Springer, Heidelberg, Aug. 2012. 3

[14] A. Joux. Authentication failures in NIST version of GCM, 2006. Comments submitted to NIST modes of operation process, `https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/comments/800-38-series-drafts/gcm/joux_comments.pdf`. 4

[15] T. Krovetz and P. Rogaway. The software performance of authenticated-encryption modes. In A. Joux, editor, *FSE 2011*, volume 6733 of *LNCS*, pages 306–327. Springer, Heidelberg, Feb. 2011. 1, 3

[16] D. McGrew. An interface and algorithms for authenticated encryption. IETF Network Working Group, RFC 5116, January 2008. 1, 2

[17] D. A. McGrew and J. Viega. The security and performance of the Galois/counter mode (GCM) of operation. In A. Canteaut and K. Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 343–355. Springer, Heidelberg, Dec. 2004. 1, 3, 5

[18] C. Namprempre, P. Rogaway, and T. Shrimpton. AE5 security notions: Definitions implicit in the CAESAR call. Cryptology ePrint Archive, Report 2013/242, 2013. `https://eprint.iacr.org/2013/242`. 5

[19] C. Namprempre, P. Rogaway, and T. Shrimpton. Reconsidering generic composition. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 257–274. Springer, Heidelberg, May 2014. 3, 4, 5

[20] T. Peyrin and Y. Seurin. Counter-in-tweak: Authenticated encryption modes for tweakable block ciphers. In M. Robshaw and J. Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 33–63. Springer, Heidelberg, Aug. 2016. 3

[21] P. Rogaway. Authenticated-encryption with associated-data. In V. Atluri, editor, *ACM CCS 2002*, pages 98–107. ACM Press, Nov. 2002. 1, 2

[22] P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In P. J. Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 16–31. Springer, Heidelberg, Dec. 2004. 1

[23] P. Rogaway. Nonce-based symmetric encryption. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 348–359. Springer, Heidelberg, Feb. 2004. 1

[24] P. Rogaway. The evolution of authenticated encryption. Real World Cryptography Workshop, Stanford, January 2013. `https://crypto.stanford.edu/RealWorldCrypto/slides/phil.pdf`. 2

[25] P. Rogaway, M. Bellare, J. Black, and T. Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In M. K. Reiter and P. Samarati, editors, *ACM CCS 2001*, pages 196–205. ACM Press, Nov. 2001. 1, 3

[26] P. Rogaway and T. Shrimpton. A provable-security treatment of the key-wrap problem. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390. Springer, Heidelberg, May / June 2006. 2, 4, 5

[27] S. Vaudenay and D. Vizár. Under pressure: Security of caesar candidates beyond their guarantees. Cryptology ePrint Archive, Report 2017/1147, 2017. `https://eprint.iacr.org/2017/1147`. 4

[28] H. Wu and B. Preneel. AEGIS: A fast authenticated encryption algorithm. In T. Lange, K. Lauter, and P. Lisonek, editors, *SAC 2013*, volume 8282 of *LNCS*, pages 185–201. Springer, Heidelberg, Aug. 2014. 3