BNymble (a short paper)
More anonymous blacklisting at almost no cost

Peter Lofgren and Nicholas Hopper

University of Minnesota

Abstract. Anonymous blacklisting schemes allow online service providers to prevent future anonymous access by abusive users while preserving the privacy of all anonymous users (both abusive and non-abusive). The first scheme proposed for this purpose was Nymble, an extremely efficient scheme based only on symmetric primitives; however, Nymble relies on trusted third parties who can collude to de-anonymize users of the scheme. Two recently proposed schemes, Nymbler and Jack, reduce the trust placed in these third parties at the expense of using less-efficient asymmetric crypto primitives. We present BNymble, a scheme which matches the anonymity guarantees of Nymbler and Jack while (nearly) maintaining the efficiency of the original Nymble. The key insight of BNymble is that we can achieve the anonymity goals of these more recent schemes by replacing only the infrequent "User Registration" protocol from Nymble with asymmetric primitives. We prove the security of BNymble, and report on its efficiency.

1 Introduction

Anonymity networks like Tor [4] and JonDo [5] allow users to access online services while concealing the parties to any particular communication, by relaying this information through several intermediaries. While these networks are an important tool for circumventing online censorship and protecting freedom of speech, they are also a "mixed blessing" for the providers of online services. In particular, while anonymous access can expand the range of users that are able or willing to contribute to an online service, it can also allow misbehaving users to abuse the online service in a way that makes it difficult to hold them accountable. As a result, several service providers – including Wikipedia and Slashdot – have chosen to block contributions from known anonymity providers, despite the potential loss of interesting contributions.

To address this problem, Johnson *et al.* [9] (inspired by [8]) proposed the notion of an *anonymous blacklisting scheme*, which allows service providers (SPs) to maintain a "blacklist" such that non-abusive users can access the service anonymously; while users on the blacklist cannot access the service, but remain anonymous. Anonymous blacklisting schemes would allow SPs to benefit from anonymous contributions and simultaneously limit abuse.

The first such construction was Nymble [9,15,14]. Nymble constructs unlinkable authentication token sequences using hash chains. A pair of Trusted Third

Parties (TTPs), the Nymble Manager (NM) and Pseudonym Manager (PM), help SPs to link future tokens from abusive users so their access can be blocked. Unfortunately, these TTPs can easily collude to de-anonymize any user.

Since the proposal of Nymble, several schemes have attempted to improve on this trust requirement. On one end, schemes such as BLAC [11,12] and EPID [2] support anonymous blacklisting of misbehaved users with no TTP. In these schemes, SPs simply add authentication tokens associated with misuse to a blacklist. When a user produces a new authentication token, she must then prove that each token on the blacklist is not linked to her new token, requiring the SP to perform a modular exponentiation for each blacklist element for every access. PEREA [13] improved on this, reducing the cost of each authentication to $O(k)$ modular exponentiations, by having each user prove that each of its last k tokens are not in a cryptographic accumulator of blacklisted tokens.

On the other hand, recent schemes such as Nymbler [7] and Jack [10] retain the TTPs from Nymble, while preventing colluding TTPs from fully de-anonymizing users. These schemes replace the symmetric primitives in Nymble with asymmetric primitives, essentially removing the dependence on blacklist size in exchange for weaker anonymity guarantees compared with BLAC. However, because they replace symmetric with asymmetric primitives, the cost of authentication and/or linking in these schemes are significantly higher than in the original Nymble.

Our Contributions. We start with a key insight: the attack that Nymbler and Jack prevent is collusion between the Pseudonym Manager and the SP or NM. Fortunately, the protocols involving the PM are the least frequently invoked, so their cost can be increased with comparatively little effect on the overall cost of authentication. We replace the Nymble PM's linkable pseudorandom function with an information-theoretically unlinkable blind signature, while leaving the rest of Nymble unchanged. The resulting scheme, which we call BNymble, provides the same anonymity guarantees as Jack and Nymbler while preserving the lower cost of authentication and linking from Nymble. We report on experiments with a prototype implementation of BNymble, showing that the total cost of authentication increases by as little as 11% over Nymble, and compare this with the higher costs of Nymbler and Jack.

2 Background and related work

Nymble. Nymble [9,15,14] was the first anonymous blacklisting scheme to appear in the literature. In Nymble, in addition to the SP and the user, there are two Trusted Parties, the Pseudonym Manager (PM) and the Nymble Manager (NM). Nymble uses an authenticated symmetric encryption scheme E, a pseudorandom function F, a message authentication code MAC and two cryptographic hash functions (modeled as random oracles), f and g; there are two secret keys KP and KN known to the PM and NM, respectively, additionally, the MAC key κpn is shared by the PM and NM and MAC Key κns is shared by the NM and SP. Nymble divides time into "linkability windows," during which a user's actions can be linked together and these are then divided further into w "time periods". Each user is assumed to have some unique identity, uid. Tsang *et al.* [15,14,9] suggest 24-hour windows, 5-minute periods, and IP address uids.

At the beginning of each linkability window d, the user connects directly to the PM to request a pseudonym $\rho = F_{KP}(d, uid), \tau = MAC_{\kappa pn}(\rho)$. The user then connects anonymously to the NM, sending ρ, τ; if the tag τ is correct, the NM forms a sequence of $w + 1$ *seeds* $s_0 = F_{KN}(\rho, d)$, $s_i = f(s_{i-1})$; *tokens* $t_i = g(s_i)$; and *ciphertexts* $c_i = E_{KN}(t_0, s_i)$. The NM gives the user *nymbles* $\nu_i = (i, t_i, c_i, MAC_{\kappa ns}(i, t_i, c_i))$. Then at the i-th time period, the user connects anonymously to the SP, checks that t_0 is not blacklisted, and provides ν_i; the SP grants access if the MAC tag is correct and t_i is not blacklisted. To complain, the SP sends ν_i to the NM, who decrypts c_i to get t_0, s_i, and computes $t_{i+1}, \ldots, t_w$ and sends these and the"canonical nymble" t_0 to the SP to add to the blacklist.

Collusion of TTPs. There are four possible collusive scenarios between a PM, NM and SP. First, the PM and NM can collude to learn which users connect to which SPs. Second, the NM and SP can collude to link all of a user's actions within a single linkability window. Third, the PM, NM, and SP can all collude together to deanonymize all of the user's activities, across linkability windows. The final scenario, involving the PM and SP, is not a privacy threat in Nymble.

Nymbler. In Nymbler [7], the PM is replaced by a Credential Manager (CM), who issues an anonymous credential on a secret x_{uid} to each user. The user then uses this credential to create his own series of seeds and tokens, with $s_0 = h^{x_{uid}}$ using $f(x) = x^2 \bmod n$, and $g(x) = \gamma^x$ over a trapdoor discrete logarithm group chosen by the NM. The user obtains blind signatures $\sigma_1, \ldots, \sigma_w$ on the tokens $t_1, \ldots, t_w$ from the NM, using efficient zero-knowledge proofs to show that they are correctly formed. The SP, on receiving $\nu_i = (t_i, \sigma_i)$ can check the signature, and the NM can extract a seed from $t_i = \gamma^{s_i}$ by computing the discrete logarithm (a costly but feasible computation using the trapdoor). The use of blind signatures prevents the NM and CM from colluding to link users to SPs; the use of anonymous credentials prevents the NM, CM, and SP from colluding to de-anonymize users.[1]

Jack. Jack [10] follows Nymbler in replacing the PM with a CM that issues credentials on a secret x_{uid}. The user creates her own nymbles by encrypting a pseudonym $h^{x_{uid}}$ under the NM's public key; the SP maintains a cryptographic accumulator of blacklisted pseudonyms. When the user connects to the SP, she presents her encrypted pseudonym along with a proof of correctness — the pseudonym corresponds to the x_{uid} in her credential, is encrypted correctly, and is not in the accumulator. To block a user, the NM decrypts the pseudonym and the SP adds it to the accumulator. As in Nymbler, the use of anonymous credentials prevents deanonymization or linking across linkability windows, and since the user creates nymbles noninteractively, the NM and CM cannot collaborate to link users to SPs.

[1] We note that [7] discuss generating x_{uid} so that it is not secret to the CM. In this case the CM and SP can collude to deanonymize users, so [6] suggests distributing the CM so that collusion between at least k CM agents and the SP is required.

Blind Signatures. BNymble uses Chaum's blind signature scheme [3]. In this scheme, the signer has public key N, an RSA modulus, and secret key $d = 3^{-1} \bmod \phi(N)$. We utilize a cryptographic hash function $H : \mathcal{M} \rightarrow \mathbb{Z}_N^*$, modeled as a random oracle. When a user wishes to obtain a blinded signature on the message $x \in \mathcal{M}$, she picks $r \in_R \mathbb{Z}_N^*$, and hands $\beta = H(x)r^3 \bmod N$ to the signer, who returns $\zeta = \beta^{1/3} \bmod N = H(x)^{1/3}r \bmod N$. Finally, the user computes $\sigma = \zeta/r = H(x)^{1/3} \bmod N$. It is easy to see that signing transcripts (β, ζ) are information-theoretically unlinkable to the signatures $(x, H(x)^{1/3} \bmod N)$; Bellare *et al.* [1] prove that it is infeasible to create $n + 1$ valid signatures from n queries under the *one-more RSA inversion problem.*

3 BNymble Protocol

Overview. In BNymble, we modify the User Registration protocol and the Nymble Acquisition protocol. In each linkability window, a user Alice first connects directly to the PM and demonstrates control over her IP address or other limited resource. She also chooses a random "blind nym" (bnym) and blinds it for signing. The PM records her *uid* (IP address) and if a signature has not already been issued for that *uid* in that linkability window, the PM signs and returns her bnym. Alice then unblinds her bnym. In the nymble acquisition phase, she opens an anonymous connection to the NM and presents her signed bnym. If the signature is valid, the NM computes seed $s_0 = F_{KN}(bnym)$ and proceeds as before. We now describe this procedure in more detail.

System Setup. In addition to the setup in Nymble, at the beginning of each linkability window i the PM chooses an RSA modulus N for signing bnyms and transmits (i, N) to the NM via an authenticated channel. For each linkability window, the PM clears the set of used IP addresses. The system includes a cryptographic hash function $H : \mathcal{M} \rightarrow \mathbb{Z}_N^*$, modeled as a random oracle.[2]

User Registration. Alice obtains a blind nym as follows:

1. Alice downloads the PM's public key for the current linkability window, N, and prepares a bnym for signing by choosing a random message $x \in_R \mathcal{M}$ and a blinding factor $r \in_R \mathbb{Z}_N^*$ and then computing $\beta = H(x)r^3 \bmod N$.
2. Alice connects directly to the PM and transmits β for signing. The PM verifies that her IP address has not previously been used this window, and then responds with $\zeta = \beta^{1/3} \bmod N = H(x)^{1/3}r \bmod N$.
3. Alice unblinds the signature by computing $\sigma = \zeta/r = H(x)^{1/3} \bmod N$.

Credential Acquisition. Alice obtains nymbles for window d as follows:

1. Alice connects anonymously to the NM and presents her bnym $(x, \sigma = H(x)^{1/3} \bmod N)$.

[2] Note that if N is a λ-bit RSA modulus, and $H' : \mathcal{M} \rightarrow \{0,1\}^{k+\lambda}$ is a random oracle, then $H(x) = H'(x) \bmod N$ will be $O(2^{-k})$-statistically close to the required oracle.

2. The NM verifies that $\sigma = H(x)^{1/3} \bmod N$. The NM computes the sequence of $w + 1$ *seeds* $s_0 = F_{KN}(x, \sigma, d)$, $s_i = f(s_{i-1})$; *tokens* $t_i = g(s_i)$; and *ciphertexts* $c_i = E_{KN}(t_0, s_i)$.
3. The NM gives the user *nymbles* $\nu_i = (i, t_i, c_i, MAC_{\kappa ns}(i, t_i, c_i))$.

The remaining Nymble protocols are identical to those described in [14].

4 Evaluation

4.1 Security Analysis

BNymble preserves Nymble's security properties: *Blacklistability*, *Rate-limiting*, *Non-frameability* and *Anonymity*, assuming the *one-more RSA inversion problem* [1] is computationally intractable.

Blacklistability. An honest Pseudonym Manager will only issue one bnym per user. Thus for a coalition of c users to authenticate after all have been blacklisted, they would either have to forge a bnym, violating the assumed intractability of the *one-more RSA inversion problem*, or they would have to break blacklistability using only c pseudonyms, violating the blacklistability of Nymble.

Non-Frameability. Since distinct users have distinct *uid*s, an honest PM will only refuse to grant a bnym to a user if that user has already received a bnym in that linkability window. Also, an honest NM will grant a different set of nymbles to each bnym. Thus there is no way for one user to frame another without violating the non-frameability of Nymble.

Anonymity. Anonymity in [9,15,14] is defined with respect to SPs only (that is, assuming non-colluding PM and NM). It is easy to see that since the nymbles in BNymble are generated according to the same process, the same property holds. We also can define anonymity in a much stronger sense: let the adversary control the PM, NM, and SP, and choose two users U and V. We allow the adversary to ask each user to register and acquire nymbles for any linkability window and any SP of the adversary's choosing, for any number k of window/SP pairs. The adversary then specifies a single, new linkability window; U and V execute the user registration protocol (with the adversary), and then execute the credential acquisition protocol in a random ordering. The adversary wins if he can guess whether U or V acquired nymbles first. The protocol is *anonymous* if no adversary can win with probability non-negligibly greater than 1/2. (Notice that since the adversary sees the nymbles issued, this implies that for any time period, the nymbles themselves are also indistinguishable.) Because bnyms are information-theoretically independent of both *uid*s and bnyms from other windows, every adversary wins this game with probability exactly 1/2.

4.2 Efficiency

In order to compare the cost of the various TTP-based anonymous blacklisting systems, we measured the costs of the basic cryptographic operations required

	Nymble	BNymble	Jack	Nymbler
User Registration (ms)	0.0008	0.70	9.12	9.12
Nymble acquisition (ms)	0.0027	0.0027	264	649
Nymble verification (ms)	0.0006	0.0006	208	0.0011

Table 1. Cost of cryptographic operations in each "nymble-like" anonymous blacklisting system. Nymble acquisition and verification costs are per nymble. All times measured on a 2.67GHz quad-core Xeon W3520 with 12GB RAM.

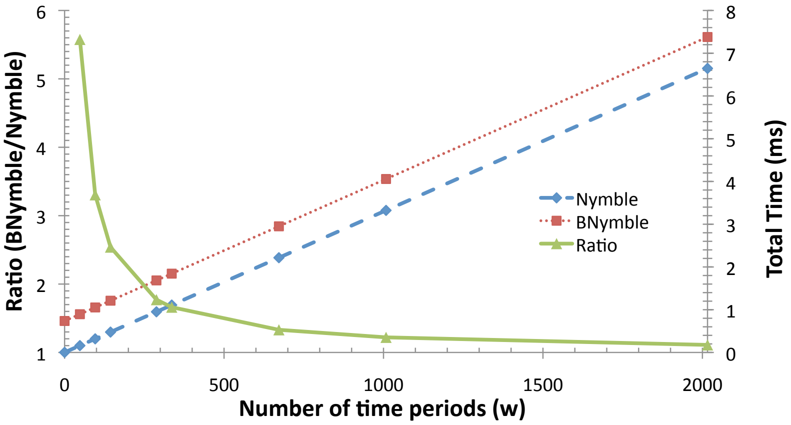

Fig. 1. Total cryptographic cost of user registration, nymble acquisition and nymble verification as a function of number of time periods per linkability window. With one week linkability windows and 5-minute time periods, the total cost of BNymble is only 11% higher than Nymble.

of the users, NM, PM, and SP in each of the systems. Table 1 shows these costs. User registration in BNymble is obviously the most expensive phase, but it is also the least executed protocol - occurring once per linkability window. Figure 1 shows how this one-time cost compares to the total cost of authentication for various linkability window sizes. At $w = 288$, as suggested by [9], The total cost of authentication in BNymble is less than a factor of 2 greater than Nymble, compared to 5 orders of magnitude from Nymbler and Jack. Longer linkability windows decrease this difference further - with 5-minute time periods and a one-week linkability window ($w = 2016$), the difference is only 11%.

5 Extensions and Future work

Coin Recovery. In Nymble, a user's nymbles for a given linkability window are a deterministic function of his IP address and the four secret keys. This means that if a user loses his nymbles, or another user with the same IP address wishes to authenticate anonymously, he can repeat the user registration and nymble acquisition protocols and get the same chain a second time. Because bnyms in BNymble are randomized and chosen by users, a literal implementation cannot support this feature. However, we can allow the user the option to choose his bnym and blinding factor pseudorandomly, based on the hardened cryptographic hash of a strong password and the index of the current linkability window. The PM would then be modified so that when a client with the same IP address requests a second bnym for the same linkability period, the blinded signature from the first request is returned, allowing the client to recover his bnym. (Since blinded signatures are information-theoretically unlinkable there is no privacy risk in doing so.)

We note that Nymble can also support "fate-sharing" of multiple users behind a Network Address Translator (NAT) based on the deterministic nature of its pseudonyms. We leave the extension of BNymble to handle this case as an important question for future work.

Identity Logging. We note that, in contrast to other "Nymble-like protocols," the BNymble PM is required to keep a log of *uid*s (IP addresses) to which a bnym has been issued for each linkability window. This is obviously undesirable. While traditional approaches to limiting the usefulness of this log can be applied,[3] these approaches do not help if the PM is compromised. Some protection can be obtained by introducing a *List Manager*, who computes an RSA key pair (M, d). User registration then becomes a slightly longer interaction: the user first connects directly to the PM, and sends a blinded signature request. The PM responds with $x = F_K(uid, d)$. The user connects anonymously to the LM, sends x and receives $y = x^{1/3} \bmod M$, and sends y to the PM. The PM checks that $y^3 = F_K(uid, d)$, and if it is, verifies that $hash(y)$ is not in the log. If successful, the PM returns the blinded signature ζ and adds $hash(y)$ to the log.

Extended Blacklisting. We note that, using the previous scheme to store (protected) lists of active *uid*s per linkability window, and using the first component of the $bnym$, $x \in \mathcal{M}$ as the "canonical nymble," BNymble can support extended blacklisting using the same techniques in [6], except that when a uid was not present during a linkability window with a non-empty blacklist, we can have the PM issue a random bnym for the window without updating the log.

Resisting Traffic Analysis. One potential concern in BNymble is side channels based on timing information: the times of registration, nymble acquisition, and first use of a service are likely to be correlated. (We note that a somewhat similar problem exists in Nymbler: after the user obtains a credential for her IP, she (anonymously) contacts the NM and sends the value h corresponding to the SP she wishes to obtain service from.) To minimize the impact of this side channel, we recommend that users first entering the system compute a random delay $\varDelta$ and wait $\varDelta$ minutes after registration and before nymble acquisition. Additionally, the PM should allow users to obtain bnyms for linkability period $d+1$ during the last half of linkability period w. Users that perform this advance registration will be indistinguishable and will help to provide cover traffic for the newly registered users.

Acknowledgments. We thank Roger Dingledine, Ian Goldberg, Ryan Henry, Apu Kapadia, and Zi Lin for helpful discussions about this paper. This work was

[3] for example, replace *uid* with $F_K(uid)$ and discarding K after the linkability window

supported by NSF grants 0546162 and 0917145 and the University of Minnesota Undergraduate Research Opportunities Program.

References

1. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The one-more-rsa-inversion problems and the security of chaum's blind signature scheme. J. Cryptology 16(3), 185–215 (2003)
2. Brickell, E., Li, J.: Enhanced privacy id: a direct anonymous attestation scheme with enhanced revocation capabilities. In: WPES '07: Proceedings of the 2007 ACM workshop on Privacy in electronic society. pp. 21–30. ACM, New York, NY, USA (2007)
3. Chaum, D.: Security without identification: transaction systems to make big brother obsolete. Commun. ACM 28(10), 1030–1044 (1985)
4. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: SSYM'04: Proceedings of the 13th conference on USENIX Security Symposium. pp. 21–21. USENIX Association, Berkeley, CA, USA (2004)
5. GmbH, J.: Jondonym: Private and secure web surfing (September 2010), `http://anonymous-proxy-servers.net/`
6. Henry, R., Goldberg, I.: Extending nymble-like systems. Tech. Rep. Technical Report CACR 2010-23, Unviersity of Waterloo (2010)
7. Henry, R., Henry, K., Goldberg, I.: Making a nymbler nymble using verbs. Tech. rep., University of Waterloo Technical Report CACR 2010-05 (2010)
8. Holt, J.E., Seamons, K.E.: Nym: Practical pseudonymity for anonymous networks. Tech. Rep. 4, BYU Internet Security Research Lab (2006)
9. Johnson, P.C., Kapadia, A., Tsang, P.P., Smith, S.W.: Nymble: Anonymous IP-address blocking. In: Proceedings of The Seventh International Symposium on Privacy Enhancing Technologies (PET), Ottawa, Canada. LNCS, vol. 4776, pp. 113–133. Springer-Verlag (June 2007)
10. Lin, Z., Hopper, N.: Jack: Scalable accumulator-based nymble system. In: WPES2010: Proceedings of the 9th ACM Workshop on Privacy in the Electronic Society. ACM (2010)
11. Tsang, P.P., Au, M.H., Kapadia, A., Smith, S.W.: Blacklistable anonymous credentials: blocking misbehaving users without ttps. In: CCS '07: Proceedings of the 14th ACM conference on Computer and communications security. pp. 72–81. ACM, New York, NY, USA (2007)
12. Tsang, P.P., Au, M.H., Kapadia, A., Smith, S.W.: BLAC: Revoking Repeatedly Misbehaving Anonymous Users Without Relying on TTPs. Tech. rep., Dartmouth Computer Science TR2008-635 (2008)
13. Tsang, P.P., Au, M.H., Kapadia, A., Smith, S.W.: Perea: Towards practical ttp-free revocation in anonymous authentication. In: CCS '08: Proceedings of the 14th ACM conference on Computer and communications security. pp. 333–344. ACM (2008)
14. Tsang, P.P., Kapadia, A., Cornelius, C., , Smith, S.W.: Nymble: Blocking Misbehaving Users in Anonymizing Networks. IEEE Transactions on Dependable and Secure Computing (TDSC) (Sep 2009)
15. Tsang, P.P., Kapadia, A., Cornelius, C., Smith, S.W.: Nymble: Blocking misbehaving users in anonymizing networks. Tech. rep., Dartmouth Computer Science TR2008-637 (2008)