Journal of Computer Applications ISSN: 0974 – 1925,
Volume-5, Issue EICA2012-4, February 10, 2012

Blocking Hacking Users in Anonymizing Network Using Blacklisting

Jenifer A
M.E Computer Science,
Anna University,GKM College of Engineering,
Chennai, India, 9600126345,
E-mail: jeni.8819@gmail.com

SankariC
Department of Computer Science and Engineering,
Anna University, GKM College of Engineering,Chennai, India
E-mail : san_nad3@yahoo.co.in

Abstract- The main aim of this project is blocking the misbehaving users who access the blocked websites through AnonymizingNetworks.In this Network, many people misbehaving the popular websites like face book, orkut, hi5, twitter, etc…through hide the client's IP address. But, website administrator routinely rely on IP address blocking for disabling Access to misbehaving users, since web site administrators cannot blacklist individual malicious users IP addresses, they blacklist the entire anonymizing network. Such measures eliminate malicious activity through anonymizing networks at the cost of denying anonymous access to behaving users. But blocking IP address is not practical if the misbehavior routes through an anonymizing network. So, a new system is proposed to solve this problem using Nymble. Nymble system, in which servers can "blacklist" misbehaving users, thereby blocking users without compromising their anonymity using through the Nymble Manager ,Pseudonym Manager. And this Nymble System accurately finding the Misbehaving users in Anonymizing networks and also maintained the blacklisted users details in server.

Keywords: *Anonymous blacklisting, privacy, revocation*

I. INTRODUCTION

ANONYMIZING networks such as Tor route traffic through independent nodes in separate administrative domains to hide a client's IP address.. Unfortunately, some users have misused such networks under the cover of anonymity, users have repeatedly defaced popular Web sites such as Wikipedia Since Web site administrators cannot blacklist individual malicious users' IP addresses, they blacklist the entire anonymizing network. Such measures eliminate malicious activity through anonymizing networks at the cost of words a few "bad apples" can spoil the fun for all.(This has happened repeatedly with Tor.1)There are several solutions to this problem, each providing some degree of accountability. In pseudonymous credential systems users log into Web sites using pseudonyms, which can be added to a blacklist if a user misbehaves. Unfortunately, this approach results in pseudonymity for all users, and weakens the anonymity provided by the anonymizing network.

Anonymous credential systems employ group signatures. Basic group signatures allow servers to revoke a misbehaving user's anonymity by complaining to a group manager. Servers must query the group manager for every authentication, and thus, lacks scalability.

Traceable signatures allow the group manager to release a trapdoor that allows all signatures generated by a particular user to be traced; such an approach does not provide the backward unlinkability that we desire, where a user's accesses before the complaint remain anonymous. Backward unlinkability allows for what we call subjective blacklisting, where servers can blacklist users for whatever reason since the privacy of the blacklisted user is not at risk. In contrast, approaches without backward unlinkability need to pay careful attention to when and why a user must have all their connections linked, and users must worry about whether their behaviors will be judged fairly. Subjective blacklisting is also better suited to servers such as Wikipedia, where misbehaviors such as questionable edits to a Webpage, are hard to define in mathematical terms. In some systems, misbehavior can indeed be defined precisely. For instance, double spending of an "e-coin" is considered a misbehavior in anonymous e-cash systems following which the offending user is deanonymized. Unfortunately, such systems work for only narrow definitions of misbehavior it is difficult to map more complex notions of misbehavior onto "double spending" or related approaches . With the dynamic accumulators a revocation operation results in a new accumulator and public parameters for the group, and all other existing users' credentials must be updated, making it impractical. Verifier-local revocation (VLR)fixes this shortcoming by requiring the server ("verifier") to perform only local updates during revocation. Unfortunately, VLRrequires heavy computation at the server that is linear in the size of the blacklist. For example, for a blacklist with 1,000 entries, each authentication would take tens of seconds,2 a prohibitive cost in practice. In contrast, our scheme takes the server about one millisecond per authentication, which is several thousand times faster than VLR. We believe these low overheads will incentivize servers to adopt such a solution when weighed against the potential benefits of anonymous publishing (e.g., whistle-blowing, reporting, anonymous tip lines, activism, and so on.).

II. PROPOSED WORK

We present a secure system called Nymble, which provides all the following properties: anonymous authentication, backward unlinkability, subjective blacklisting, fast authentication speeds, rate limited anonymous connections, revocation auditability (where users can verify whether they have been blacklisted), and also addresses the Sybil attack to make its deployment practical. In Nymble, users acquire an ordered collection of nymbles, a special type of pseudonym, to connect to Websites. Without additional information, these nymbles are computationally hard to link,4 and hence, using the stream of nymbles simulates anonymous access to services. Web sites, however, can blacklist users by obtaining a seed for a particular nymble, allowing them to link future nymbles from the same user—those used before the complaint remains unlinkable. Servers can therefore blacklist anonymous users without knowledge of their IP addresses while allowing behaving users to connect anonymously. Our system ensures that users are aware of their blacklist status before they present a nymble, and disconnect immediately if they are blacklisted. Although our work applies to anonymizing networks in general, we consider Tor for purposes of exposition. In fact, any number of anonymizing networks can rely on the same

Nymble system, blacklisting anonymous users regardless of their anonymizing network(s) of choice.

A. An Overview to Nymble

We now present a high-level overview of the Nymble system, and defer the entire protocol description and security analysis to subsequent sections.

B. Resource-Based Blocking

To limit the number of identities a user can obtain (called the Sybil attack [19]), the Nymble system binds nymbles to resources that are sufficiently difficult to obtain in great numbers. For example, we have used IP addresses as the resource in our implementation, but our scheme generalizes to other resources such as email addresses, identity certificates, and trusted hardware. We address the practical issues related with resource-based blocking in Section 8, and suggest other alternatives for resources. We do not claim to solve the Sybil attack. This problem is faced by any credential system and we suggest some promising approaches based on resource-based blocking since we aim to create a real-world deployment.

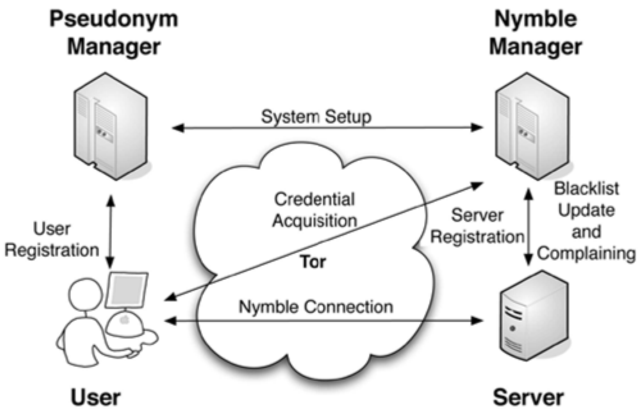

Figure 1: The Nymble system architecture showing the various modes of interaction. Note that users interact with the NM and servers though the anonymizing network.

C. The Pseudonym Manager

The user must first contact the Pseudonym Manager (PM) and demonstrate control over a resource; for IP-address blocking, the user must connect to the PM directly (i.e., not through a known anonymizing network), as shown in Fig.1.We assume the PM has knowledge about Tor routers, for example, and can ensure that users are communicating with it directly.6 Pseudonyms are deterministically chosen based on the controlled resource, ensuring that the same pseudonym is always issued for the same resource.

Note that the user does not disclose what server he or she intends to connect to, and the PM's duties are limited to mapping IP addresses (or other resources) to pseudonyms. As we will explain, the user contacts the PM only once per linkability window (e.g., once a day).

D. The Nymble Manager

After obtaining a pseudonym from the PM, the user connects to the Nymble Manager (NM) through the anonymizing network, and requests nymbles for access to a particular server (such as Wikipedia). A user's requests to the NM are therefore pseudonymous, and nymbles are generated using the user's pseudonym and the server's identity. These nymbles are thus specific to a particular user-server pair. Nevertheless, as long as the PM and the NM do not collude, the Nymble system cannot identify which user is connecting to what server; the NM knows only the pseudonym-server pair, and the PM knows only the user identity-pseudonym pair. To provide the requisite cryptographic protection and security properties, the NM encapsulates nymbles within nymble tickets. Servers wrap seeds into linking tokens, and therefore, we will speak of linking tokens being used to link future nymble tickets. The importance of these constructs will become apparent as we proceed.

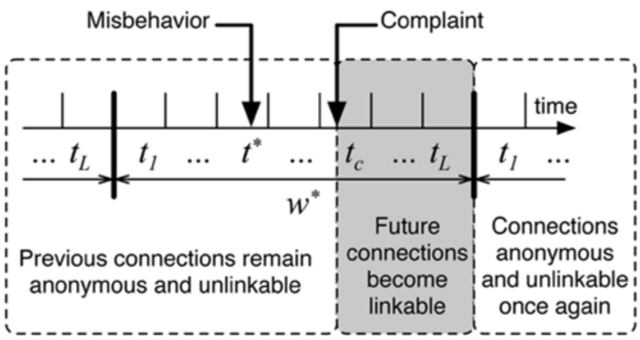

Figure 2: The life cycle of a misbehaving user. If the server complains in time period tc about a user's connection in t_, the user becomes linkable starting in tc. The complaint in tc can include nymble tickets from only tc_1 and earlier.

anonymizing network, and requests nymbles for access to a particular server (such as Wikipedia). A user's requests to the NM are therefore pseudonymous, and nymbles are generated using the user's pseudonym and the server's identity. These nymbles are thus specific to a particular user-server pair. Nevertheless, as long as the PM and the NM do not collude, the Nymble system cannot identify which user is connecting to what server; the NM knows only the pseudonym-server pair, and the PM knows only the user identity-pseudonym pair. To provide the requisite cryptographic protection and security properties, the NM encapsulates nymbles within nymble tickets. Servers wrap seeds into linking tokens, and therefore, we will speak of linking tokens being used to link future nymble tickets. The importance of these constructs will become apparent as we proceed.

E. **Time**

Nymble tickets are bound to specific time periods. As illustrated in Fig. 2, time is divided into linkability windows of duration W, each of which is split into L time periods of duration T (i.e., W ¼L _ T). We will refer to time periods and linkability windows chronologically as t1; t2; . . . ; tL and w1; w2; . . . , respectively. While a user's access within a time period is tied to a single nymble ticket, the use of different nymble tickets across time periods grants the user anonymity between time periods. Smaller time periods provide users with higher rates of anonymous authentication, while longer time periods allow servers to rate-limit the number of misbehaviors from a particular user before he or she is blocked. For example, T could be set to five minutes, and W to one day (and thus, L ¼ 288). The linkability window allows for dynamism since resources such as IP addresses can get reassigned and it is undesirable to blacklist such resources indefinitely, and it ensures forgiveness of misbehavior after a certain period of time. We assume all entities are time synchronized (for example, with time.nist.gov via the Network Time Protocol (NTP)), and can thus calculate the current linkability window and time period.

F. **Blacklisting a User**

If a user misbehaves, the server may link any future connection from this user within the current linkability window (e.g., the same day). Consider Fig. 2 as an example: A user connects and misbehaves at a server during time period t_ within linkability window w_. The server later detects this misbehavior and complains to the NM in time period tc (t_ <tc _ tL) of the same linkability window w_. As part of the complaint, the server presents the nimble ticket of the misbehaving user and obtains the corresponding seed from the NM. The server is then able to link future connections by the user in time periods tc; tc þ 1; . . . ; tL of the same linkability window w_ to the complaint. Therefore, once the server has complained about a user, that user is blacklisted for the rest of the day, for example (the linkability window). Note that the user's connections in t1; t2; . . . ; t_; t_ þ 1; . . . ; tc remain unlinkable (i.e., including those since the misbehavior and until the time of complaint). Even though misbehaving users can be blocked from making connections in the future, the users' past connections remain unlinkable, thus providing backward unlinkability and subjective blacklisting.

G. Notifying the User of Blacklist Status

Users who make use of anonymizing networks expect their connections to be anonymous. If a server obtains a seed for that user, however, it can link that user's subsequent connections. It is of utmost importance then that users be notified of their blacklist status before they present a nimble ticket to a server. In our system, the user can download the server's blacklist and verify her status. If blacklisted, the user disconnects immediately. Since the blacklist is cryptographically signed by the NM, the authenticity of the blacklist is easily verified if the blacklist was updated in the current time period (only one update to the blacklist per time period is allowed). If the blacklist has not been updated in the current time period, the NM provides servers with "daisies" every time period so that users can verify the freshness of the blacklist ("blacklist from time period told is fresh as of time period t now"). As discussed in Section 4.3.4, these daisies are elements of a hash chain, and provide a lightweight alternative to digital signatures. Usingdigital signatures and daisies, we thus ensure that race conditions are not possible in verifying the freshness of a blacklist. is guaranteed that he or she will not be linked if the user verifies the integrity and freshness of the blacklist before sending his or her nymble ticket.

H. Summary of Updates to the Nymble Protocol

We highlight the changes to Nymble since our conference paper.Previously, we had proved only the privacy properties associated with nymbles as part of a two-tiered hash chain. Here, we prove security at the protocol level. This process gave us insights into possible (subtle) attacks against privacy, leading us to redesign our protocols and refine our definitions of privacy. For example, users are now either legitimate or illegitimate, and are anonymous within these sets (see Section 3). This redefinition affects how a user establishes a “Nymble connection” (see Section 5.5), and now prevents the server from distinguishing between users who have already connected in the same time period and those who are blacklisted, resulting in larger anonymity sets.

A thorough protocol redesign has also resulted in several optimizations. We have eliminated blacklist version numbers and users do not need to repeatedly obtain the current version number from the NM. Instead servers obtain proofs of freshness every time period, and users directly verify the freshness of blacklists upon download. Based on a hashchain approach, the NM issues lightweight daisies to servers as proof of a blacklist’s freshness, thus making blacklist updates highly efficient. Also, instead of embedding seeds, on which users must perform computation to verify their blacklist status, the NM now embeds a unique identifier nymble_, which the user can directly recognize. Finally, we have compacted several data structures, especially the servers’ blacklists, which are downloaded by users in eachconnection, and report on the various sizes in detail in Section 6.Wealso report on our open-source implementation of Nymble, completely rewritten as a C library for efficiency.

III. IMPLEMENTATION AND EXPERIMENTAL RESULTS

We implemented Nymble as a C++ library along with Ruby and JavaScript bindings. One could, however, easily compile bindings for any of the languages (such as Python, PHP, and Perl) supported by the Simplified Wrapper and Interface Generator (SWIG), for example. We utilize Open SSL for all the cryptographic primitives. We use SHA-256 for the cryptographic hash functions; HMAC-SHA-256 for the message authentication MA; AES-256 in CBC-mode for the symmetric encryption Enc; and 2,048-bit RSASSA-PSA for the digital signatures Sig. We chose RSA over DSA for digital signatures because of its faster verification speed—in our system, verification occurs more often than signing. We evaluated our system on a 2.2 GHz Intel Core 2 Duo Mac book Pro with 4 GB of RAM. The PM, the NM, and the server were implemented as Mongrel (Ruby’s version of Apache) servers. The user portion was implemented as a Firefox 3 extension in JavaScript with XPCOM bindings to the Nymble C++ library. For each experiment relating to protocol performance, we report the average of 10 runs. The evaluation of data structure size the byte count of the marshaled data structures that would be sent over the network.

Figure 3.shows the size of the various data structures. TheX-axis represents the number of entries in each data structure—complaints in the blacklist update request, tickets in the credential (equal to L, the number of timeperiods in a linkability window), nymbles in the blacklist, tokens and seeds in the blacklist update response, and nymbles in the blacklist. For example, a linkability window of one day with five minute time periods equates to L ¼ 288.11 The size of a credential in this case is about 59 KB. The size of a blacklist update request with 50 complaints is roughly 11 KB, whereas the size of a blacklist update response for 50 complaints is only about 4 KB. The size of a blacklist (downloaded by users before each connection) with 500 nymbles is 17 KB. In general, each structure grows linearly as the number of entries increases. Credentials and blacklist update requests grow at the same rate because a credential is a collection of tickets which is more or less what is sent as accomplaint list when the server wishes to update its blacklist. In our implementation, we use Google's Protocol Buffers to (un)marshal these structures because it is cross-platform friendly and language-agnostic.

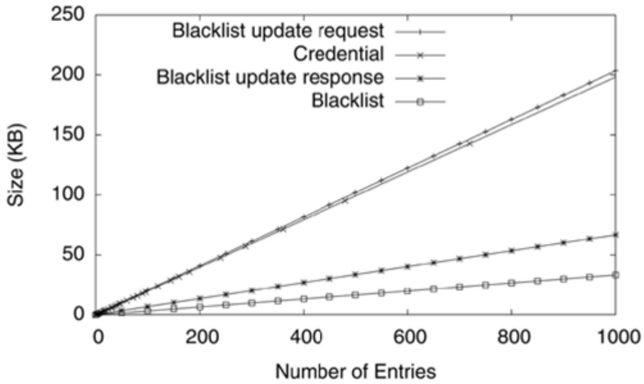

Figure 3.The marshaled size of various Nymble data structures. The X-axis refers to the number of entries—complaints in the blacklist update request, tickets in the credential, tokens and seeds in the blacklist update response, and nymbles in the blacklist.

Fig. 4a shows the amount of time it takes the NM to perform various protocols. It takes about 9 ms to create a credential when L ¼ 288. Fig. 4b shows the amount of time it takes the server and user to perform various protocols. These protocols are relatively inexpensive by design, i.e., the amount of computation performed by the users and servers should be minimal. list. If the server were to issue a blacklist update request with 500 complaints, it would take less than 3 ms for the server to update its blacklist.

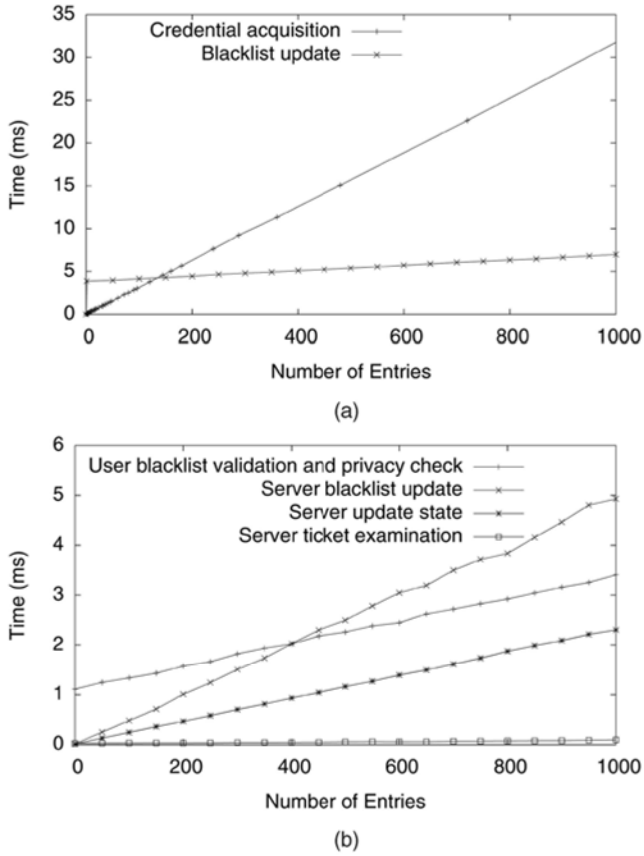

Figure 4.Nymble's performance at (a) the NM and (b) the user and the server when performing various protocols. (a) Blacklist updates take several milliseconds and credentials can be generated in 9 ms for the suggested parameter of L ¼ 288. (b) The bottleneck operation of server ticket examination is less than 1 ms and validating the blacklist takes theuser only a few ms

IV. CONCLUSIONS AND FURTHER RESEARCH

We have proposed and built a comprehensive credential system called Nymble, which can be used to add a layer of accountability to any publicly known anonymizing network. servers can blacklist misbehaving users while maintaining their privacy, and we show how these properties can be attained in a way that is practical, efficient, and sensitive to the needs of both users and We hope that our work will increase the mainstream acceptance of anonymizing networks such as Tor, which has, thus far, been completely blocked by several services because of users who abuse their anonymity.

V. ACKNOWLEDGMENT

We wish to express our sincere thanks to all the staff member of C.S.E Department, GKM College of Engineering for their help and cooperation.

REFERENCES

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, “A Practical and Provably Secure Coalition-Resistant Group Signature,Scheme,” Proc. Ann. Int’l Cryptology Conf. (CRYPTO), Springer,pp. 255-270, 2000.
2. G. Ateniese, D.X. Song, and G. Tsudik, “Quasi-Efficient Revocation in Group Signatures,” Proc. Conf. Financial Cryptography, Springer, pp. 183-197, 2002.
3. M. Bellare, R. Canetti, and H. Krawczyk, “Keying Hash Functions for Message Authentication,” Proc. Ann. Int’l Cryptology Conf. (CRYPTO), Springer, pp. 1-15, 1996.
4. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, “A Concrete Security Treatment of Symmetric Encryption,” Proc. Ann. Symp. Foundations in Computer Science (FOCS), pp. 394-403, 1997.
5. M. Bellare and P. Rogaway, “Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols,” Proc. First ACM Conf. Computer and Comm. Security, pp. 62-73, 1993.
6. M. Bellare, H. Shi, and C. Zhang, “Foundations of Group Signatures: The Case of Dynamic Groups,” Proc. Cryptographer’s Track at RSA Conf. (CT-RSA), Springer, pp. 136-153, 2005.
7. D. Boneh and H. Shacham, “Group Signatures with Verifier-Local Revocation,” Proc. ACM Conf. Computer and Comm. Security, pp. 168-177, 2004.
8. S. Brands, “Untraceable Off-Line Cash in Wallets with Observers (Extended Abstract),” Proc. Ann. Int’l Cryptology Conf. (CRYPTO), Springer, pp. 302-318, 1993.
9. E. Bresson and J. Stern, “Efficient Revocation in Group Signatures,” Proc. Conf. Public Key Cryptography, Springer, pp. 190-206, 2001.
10. J. Camenisch and A. Lysyanskaya, “An Efficient System for Non-Transferable Anonymous Credentials with Optional Anonymity Revocation,” Proc. Int’l Conf. Theory and Application of CryptographicTechniques (EUROCRYPT), Springer, pp. 93-118, 2001.
11. J. Camenisch and A. Lysyanskaya, “Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials,” Proc. Ann. Int’l Cryptology Conf. (CRYPTO), Springer, pp. 61-76,2002.
12. J. Camenisch and A. Lysyanskaya,“SignatureSchemes and Anonymous Credentials from Bilinear Maps,” Proc.Ann. Int’l Cryptology Conf. (CRYPTO), Springer,pp. 56-72, 2004.
13. D. Chaum, “Blind Signatures for Untraceable Payments,” Proc.Ann. Int’l Cryptology Conf. (CRYPTO), pp. 199-203, 1982.
14. D. Chaum, “Showing Credentials without Identification Transfeering Signatures between Unconditionally Unlinkable Pseudonyms,”Proc. Int’l Conf. Cryptology (AUSCRYPT), Springer,pp. 246-264, 1990.
15. D. Chaum and E. van Heyst, “Group Signatures,” Proc. Int’l Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT),pp. 257-265, 1991.

BIOGRAPHY

A.Jeniferreceived B.E in computer science at SRR Engineering College, Chennai. Now Pursuing M.E at GKM College of Engineering. Her research interests include Transaction on Secure and Dependable Computing.