A survey on anonymous ip address blocking

[1.] Prof. P.Pradeepkumar [2.]Amer Ahmed khan [3.]B. Kiran Kumar

ABSTRACT
These days, the Internet is full of suspicious actions and people. We always advise to not take a chance in protecting your IP Address information online.

TheIP was being shown everywhere! To advertisersandother places ,even from SPAM who compromised usersidentity. No more I said, and developed software that would hide ip address.we outline a security protocol that uses resource constrained trusted hardware to facilitate anonymous IP-address blocking in anonym zing networks such as Tor.
Tor allows users to access Internet services privately by using a series of Tor routers to obfuscate the route from the client to the server, thereby hiding the client's IP address from the server. The success of Tor, however, has been limited because of malicious users who misuse the network. administrators block all known exit nodes of anonymizing networks, denying nonymous access to misbehaving and behaving users alike.

To address this problem, we present Nymble, a system in which servers can "blacklist" misbehaving users, thereby blocking users without compromising their anonymity. Our system is thus agnostic to different servers' definitions of misbehavior servers can blacklist users for whatever reason, and the privacy of blacklisted users is maintained.
The IP-address anonymity provided by Tor, however, makes it difficult for administrators to deny access to such

offenders .As a result, administrators resort to blocking all Tor exit nodes, effectively denying
anonymous access for all Tor's users. Our solution makes use of trusted hardware and allows services like Tor to provide anonymous blocking of IP addresses while requiring only a modest amount of storage at the trusted node.

Key terms: IP , pockets, NYMBLE,Anonymizing networks, privacy.

Introduction

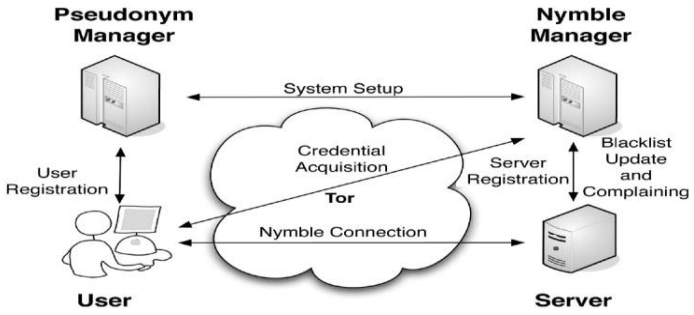
Anonymizing networks such as re-route a user's traffic between several nodes in different domains. Since these nodes are operated independently, users are able to trust the anonymizing network to provide anonymity. Real-world deployments of anonymizing networks, however, have had limited success because of their misuse

Administrators of websites are unable to blacklist malicious users' IP addresses because of their anonymity. Left with no other choice, these administrators opt to blacklist the entire anonymizing network. This approach eliminates malicious activity through such networks, but at the cost of the anonymity of honest users. In other words, a few "bad apples" can spoil the fun for everybody else using the anonymizing network. (In fact, this has happened repeatedly with To solve this problem, we present a secure protocol based on trusted hardware that allows servers to block anonymous users without knowledge of their actual IP addresses. Although this work applies to anonymizing networks in general, we consider Tor for purposes of exhibition. Building and prototyping a system based on our proposed solution is ongoingwork. In this paper we present our proposed solution and protocol.

Anonymizing networks such as Tor allow users to access Internet services privately by using a series of routers to hide the client's IP address from the server. The success of such networks, however, has been limited by users employing this anonymity for abusive purposes such as defacing popular websites.
Website administrators routinely rely on IP-address blocking for disabling access to misbehaving users, but blocking IP addresses is not practical if the abuser routes through an anonymizing network. As a result,

2 An Overview To Nymble

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



To limit the number of identities a user can obtain (called the Sybil attack), 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.

Wedonotclaimto solvethe Sybi lattack.
Thisproblemisfacedbyanycredentialsystem],andwesuggestsomepromisingapproachesbasedonresource-based blocking since we aim to create a real-world deployment.

2.2 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), 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 pseudo-nym 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).

2.3 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-serverpair. 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. Nymble tickets are bound to specific time periods, time is divided into linkability windows of durationW, each of which is split into L time periods of duration T (i.e., W ¼ L T). We will refer to time periods and linkability windows chronologically as t_1; t_2; . . . ; t_L and w_1;w_2;..., 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.Summary of Updates to the Nymble Protocol We highlight the changes to Nymble since our onferencepaper [24]. 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. This redefinition affects how a user establishes a "Nymble connection" , 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 hash chain approach, the NM issues lightweightdaisiestoservers 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 each user.

Our Nymble Construction

System Setup

During setup, the NM and the PM interact as follows:

1. The NM executes NMInitStateðÞ (see Algorithm 10) and initializes its state nmState to the algorithm's output.
2. The NM extracts $macKey_{NP}$ from nmState and sends it to the PM over a type-Auth channel. $macKey_{NP}$ is a shared secret between the NM and the PM, so that the NM can verify the authenticity of pseudonyms issued by the .
3. The PM generates $nymKey_P$ by running Mac.Key-Gen() and initializes its state pmState to the pair $ðnymKey_P ; macKey_{NP} Þ$.
4. The NM publishes $verKey_N$ innmState in a way that the users in Nymble can obtain it and verify its integrity at any time (e.g., during registration).

Server Registration

To participate in the Nymble system, a server with identity sid initiates a type-Auth channel to the NM, and registers with the NM according to the Server Registration protocol below. Each server may register at most once in any linkability window.

User Registration

In this procedure, user Alice interacts with the PM in order to register herself to the NYMBLE system for linkability window k. Alice obtains a pseudonym from the PM upon a successful termination of such an interaction. The communication channel between them is confidential and PM-authenticated.

To register, Alice authenticates herself as a user with identity id to the PM by demonstrating her control over some resource(s) as discussed, after which the PM computes pnymHkhkP (id, k) and macPNHMAChmkNP (pnym, k), and returns hpnym, macPNi to Alice, who stores it privately.

Acquisition of Nymble Tickets In order for Alice to authenticate to any server Sj during any linkability window Wk, she must present a nymble ticket to the server. The following describes how she can obtain a credential from the NM containing such tickets. The communication channel is anonymous

(e.g., through Tor), confidential and NM-authenticated.

Alice sends her hpnym, macPNi to the NM, after which the NM:

1. asserts that macPN = HMAChmkNP (pnym, k),
2. computesnymbleTKT` NymbleTktGennmsk (pnym, j, k, `), for ` = 1 to L, and
3. returns cred as hseed, nymbleTKT1, nymbleTKT2, . . . , nymbleTKTLi, where seed =

HkhkN (pnym, j, k) is the seed used within NymbleTktGen.

Alice may acquire credentials for different servers and different linkability windows at any time. She stores these credentials locally before she needs them.

Efficiency. This protocol has a timing complexity of O(L).6 All the computations are quick symmetric operations—there are two cryptographic hashes, two HMACs and one symmetric encryption per loop-iteration A credential is of size O(L).
Request for Services
At a high level, a user Alice presents to server Bob the nymble ticket for the current time period. As nymble tickets are unlinkable until servers complain against them (and thereby blacklisting the corresponding user or IP address), Alice must check whether she is on Bob's blacklist, and verify its integrity and freshness. If Alice decides to proceed, she presents her nymble ticket to Bob, and Bob verifies that the nymble ticket is not on his blacklist. Bob also retains the ticket in case he wants to later complain against the current access. For example,Wikipedia might detect a fraudulent posting several hours after it has been made. The nymble ticket associated with that request can be used to blacklist future accesses by that user.

Ticket Examination

1. The user sets ticketDisclosed in usrEntries½sid inusrState to true. She then sends hticketi to the server, where ticket is ticket½t^{now} in cred in usrEntries½sid in usrState.Note that the user discloses ticket for time period t^{now}after verifying blist's freshness for t^{now}. This procedure avoids the situation in which the user verifies the current blacklist just before a time period ends, and then presents a newer ticket for the next time period.

2. On receiving hticketi, the server reads the currenttime period and linkability window as t^{now} and^{wnow},returns false, the NM terminates with failure; it proceeds otherwise.
3.The NM runsNMCreateCredential$_{nmState}$ðpnym; sid; w_{now}Þ;which returns a credential. The NM sends cred to the user and terminates with success.
4.The user, on receiving cred, creates usrEntry:¼ðsid; cred;falseÞ.

Nymble Connection establishment

To establish a connection to a server sid, the user initiates a type-Anon channel to the server, followed by the Nymbleconnection establishment protocol described below.

5.5.1 Blacklist Validation

1. The server sends hblist; certi to the user, where blist is its blacklist for the current time period and cert is the certificate on blist. (We will describe how the server can update its blacklist soon.)
2. The user reads the current time period and linkability window as t^{now} and^{wnow} andassumes these values to be current for the rest of the protocol.
3. For freshness and integrity, the user checks if VerifyBL$_{usrState}$sid; t^{now}; wnow; blist; cert ¼ true:If not, she terminates the protocol with failure.

5.5.2 Privacy Check

Since multiple connection establishment attempts by a user to the same server within the same time period can be linkable, the user keeps track of whether she has alreadyrespectively. The server then checks that:ticket isfresh,i.e.,ticket
2slistinserver's state. ticket is valid, i.e., on input ðt^{now}; w^{now}; ticketÞ'ServerVerifyTicket returns true. Ticket is not linked ServerLinkTicket$_{svrState}$ðticketÞ ¼ false:
If any of the checks above fails, the server sendshgoodbyei to the user and terminates with failure. Otherwise, it adds ticket to slist in its state, sends hokayi to the user, and terminates with success.On receiving hokayi, the user terminates with success.
Algorithm:.ServerLinkTicketInput: ticket 2 T
Persistent state: svrState 2 S_S

Output: b 2 ftrue; falseg

1: Extract lnkng-tokens from svrState2: ð ;nymble; Þ :¼ ticket
3: for all i ¼ 1 to jlnkng-tokensj do
4: if ð ;nymbleÞ ¼ lnkng-tokens½i then
5: return true 6: return false

Conclusions

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 services.
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.

References

1. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In MihirBellare, editor, CRYPTO, volume 1880 of LNCS, pages 255–270.Springer, 2000.

2. MihirBellare, Daniele Micciancio, and BogdanWarinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, EUROCRYPT, volume 2656 of LNCS, pages 614–629. Springer, 2003.

3. MihirBellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, CT-RSA, volume 3376 of LNCS, pages 136–153. Springer, 2005.

4. Stefan Brands. Untraceable off-line cash in wallets with observers (extended abstract). In Douglas R. Stinson, editor, CRYPTO, volume 773 of LNCS, pages 302–318. Springer, 1993.

5. Jan Camenisch, Susan Hohenberger, MarkulfKohlweiss, Anna Lysyanskaya, and Mira Meyerovich. How to win the clonewars: efficient periodic n-times anonymous authentication. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, ACM Conference on Computer and Communications Security, pages 201–210. ACM, 2006.

6. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya.Compact e-cash. In Ronald Cramer, editor, EUROCRYPT, volume 3494 of LNCS, pages 302–321. Springer, 2005.

7. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, EUROCRYPT, volume 2045 of LNCS, pages 93–118.Springer, 2001.

8. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew K. Franklin, editor, CRYPTO, volume 3152 of LNCS, pages 56–72. Springer, 2004.

9. David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM, 4(2), February 1981.

10. David Chaum. Blind signatures for untraceable payments. In CRYPTO, pages 199–203, 1982.

11. David Chaum. Showing credentials without identification transfeering signatures between unconditionallyunlinkable pseudonyms. In Jennifer Seberry and Josef Pieprzyk, editors, AUSCRYPT, volume 453 of LNCS, pages 246–264. Springer, 1990.

12. David Chaum and Eug`ene van Heyst. Group signatures. In EUROCRYPT, pages 257–265, 1991.

13. Lidong Chen. Access with pseudonyms. In Ed Dawson and Jovan Dj. Golic, editors, Cryptography: Policy and Algorithms, volume 1029 of LNCS, pages 232–243.Springer, 1995
.
14. Ivan Damg°ard. Payment systems and credential mechanisms with provable security against abuse by individuals. In ShafiGoldwasser, editor, CRYPTO, volume 403 of LNCS, pages 328–335. Springer, 1988.

15. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In Usenix Security Symposium, pages 303–320, August 2004.

16. John R. Douceur. The sybil attack. In Peter Druschel, M. FransKaashoek, and Antony I. T. Rowstron, editors, IPTPS, volume 2429 of LNCS, pages 251–260. Springer, 2002.
17
17. Jason E. Holt and Kent E. Seamons.Nym: Practical pseudonymity for anonymous networks. Internet Security Research Lab Technical Report 2006-4, Brigham Young University, June 2006.

18. AggelosKiayias, YiannisTsiounis, and Moti Yung.Traceable signatures. In Christian Cachin and Jan Camenisch, editors, EUROCRYPT, volume 3027 of LNCS, pages 571–589. Springer, 2004.
19. Anna Lysyanskaya, Ronald L. Rivest, AmitSahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, Selected Areas in Cryptography, volume 1758 of LNCS, pages 184–199. Springer, 1999
.
20. NIST. FIPS 186-2: Digital signature standard (dss). Technical report, National Institute of Standards and Technology (NIST), 2000. http://csrc.nist.gov/publications/ fips/fips186-2/fips186-2-change1.pdf.

AUTHORS PROFILE

Prof Pradeep Kumar Puram
B.E M.Tech(Ph.D) having several
years of experience inAcademic & Industry.
Currently he is the Professor & Head of
Department Of Computer Science & Engineering At Vivekananda Institute of
Technology & Science, Karimnagar, he has guided many UG & PG students. His research areas of interest are Software Engineering, Data mining &Data Warehousing, Information Security, Web
Technologies

2. ——————|
Amer Ahmed khan pursuing M.TechCS
fromVivekananda Institute of Technology &Science, Karimnagar,. His

Research areas include Programming
In JAVA,Security, Cryptography &
Web Technologies currently focusing on Information estimation on Wireless Networks.

3. 

KiranKumar B

Assistant Professor,
Dept of CSE,
VITS Karimnagar.

He received M .Tech from JNTUH .
He has the sound knowledge in computer networks and assembly language programming ,expertised in C with real time application development.