IJCSC 0973-7391
International Journal of Computer Science & Communication
Volume 6, Issue 2, April-September 2015
ISSN: 0973-7391 Impact Factor: 2.5 Available at www.csjournals.com

Result of Blocking of Mischievous users in Anonymizing Networks using Nymble System

Anil Kumar, Kirti Bhatia
Deptt. of Computer Science & Applications, Sat Kabir Institute of Technology and Management, Ladrawan, Haryana, India
akumar7476@gmail.com, bhatia.kirti.it@gmail.com

ABSTRACT There are some networks called "Anonymizing Networks" which allow users to gain access to internet services without revealing their identity (IP-addresses) to the servers. Networks such as "Tor (The Onion Router)","Crowds" and "I2P" gained popularity in the years 2002-2007, but the success of such networks however has been limited by users employing this anonymity for abusive purposes such as defacing popular websites such as "Wikipedia". Website Administrators blocks entire network which is connected to the abusive system to get rid of the abuser. Hence, well-behaved users also get blocked due to this action. To address this problem, we present a Nymble system in which servers can "blacklist" mischievous users without affecting good users and also maintaining anonymity across the network.

Keywords: Anonymous, privacy, revocation, pseudonym.

1. INTRODUCTION

Networks which provide anonymity to users such as Crowds and Tor [1], [2], will route the traffic through independent nodes in separate administrative domains to hide the user's IP address. Tor network routes through several series of routers to decrease the probability of predicting the IP address of the user by the server and hence increases the anonymity. But unfortunately some users have misused such networks by taking the advantage of their anonymity to deface popular websites. Since website administrators cannot blacklist individual malicious users' IP addresses, they blacklist the entire anonymizing network. Such measures will definitely eliminate malicious activity through anonymizing networks, but at the same time it results in denial of service to behaving users as well. In other words, a poisonous fish can kill all other fishes under that same area. (This has happened repeatedly with Tor).

Next Figure show the Nymble system architecture which has various modes of interaction in the network of anonymity. This system has overcome many drawbacks which arise from the previously proposed systems including the speed, computation work, security etc.

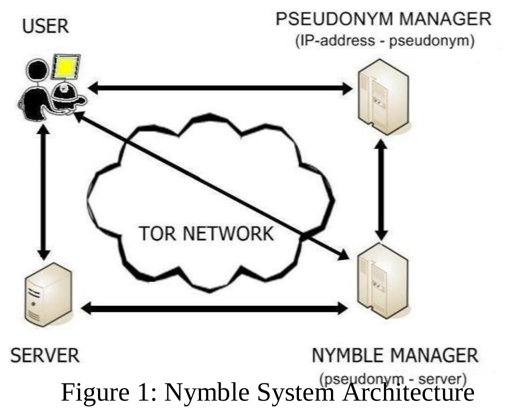

Figure 1: Nymble System Architecture

2. Materials and Methods

The notation $a \in_R S$ represents an element drawn uniformly at random from non-empty set S. N0 is the set of non-negative integers, and N is the set N0\{0}. s[i] is the i-th element of list s. s||t is the concatenation of (the unambiguous encoding of) lists s and t. The empty list is denoted by $\emptyset$. We sometimes treat lists of tuples as dictionaries. For example, if L is the list ((Alice, 1234),(Bob, 5678)), then L[Bob] denotes the tuple (Bob, 5678). If A is a (possibly probabilistic) algorithm, then A(x) denotes the output when A is executed given the input x. a := b means that b is assigned to a.

• Data structures

. Nymble uses several important data structures:

Algorithm 1 PMCreatePseudonym
```
Input: •uid•w• • • • N
Persistent state: pmState • • P
Output: pnym • •
1: Extract nymKey$_P$•macKey$_{NP}$ from pmState
2: nym •• MA•Mac•uid••w•nymKey$_P$•
3: mac •• MA•Mac•nym••w•macKey$_{NP}$•
4: return pnym •• •nym•mac•
```

Algorithm 2 NMVerifyPseudonym
```
Input: •pnym•w• • • • N
Persistent state: nmState • • N
Output: • • •true•false•
1: Extract macKey$_{NP}$ from nmState
2: •nym•mac• •• pnym
3: return mac •$\overset{?}{}$ MA•Mac•nym••w•macKey$_{NP}$•
```

Algorithm 3 NMCreateCredential
```
Input: •pnym•sid•w• • • • • • N
Persistent state: nmState • • N
Output: cred • •
1: Extract macKey$_{NS}$•macKey$_N$•seedKey$_N$•encKey$_N$
   from keys in nmState
2: seed$_0$ •• • •Mac•pnym••sid••w•seedKey$_N$••
3: nymble$^*$ •• • •seed$_0$•
4: for t from • to • do
5:     seed$_t$ •• • •seed$_{t•1}$•
6:     nymble$_t$ •• • •seed$_t$•
7:     ctxt$_t$ •• Enc•Encrypt•nymble$^*$••seed$_t$•encKey$_N$•
8:     ticket$_t^*$ •• sid••t••w••nymble$_t$••ctxt$_t$
9:     mac$_{N,t}$ •• MA•Mac•ticket$_t^*$•macKey$_N$•
10:    mac$_{NS,t}$ •• MA•Mac•ticket$_t^*$••mac$_{N,t}$•macKey$_{NS}$•
11:    tickets•t• •• •t•nymble$_t$•ctxt$_t$•mac$_{N,t}$•mac$_{NS,t}$•
12: return cred •• •nymble$^*$•tickets•
```

Algorithm 4 NMVerifyTicket
```
Input: •sid•t•w•ticket• • • • N$^2$• •
Persistent state: svrState
Output: • • •true•false•
1: Extract macKey$_N$ from keys in nmState
2: •••nymble•ctxt•mac$_N$•mac$_{NS}$• •• ticket
3: content •• sid••t••w••nymble••ctxt
4: return mac$_N$ •$\overset{?}{}$ MA•Mac•content•macKey$_N$•
```

Algorithm 5 ServerVerifyTicket
```
Input: •t•w•ticket• • N$^2$• •
Persistent state: svrState
Output: • • •true•false•
1: Extract sid•macKey$_{NS}$ from svrState
2: •••nymble•ctxt•mac$_N$•mac$_{NS}$• •• ticket
3: content •• sid••t••w••nymble••ctxt••mac$_N$
4: return mac$_{NS}$ •$\overset{?}{}$ MA•Mac•content•macKey$_{NS}$•
```

Algorithm 6 UserCheckIfBlacklisted
```
Input: •sid•blist• • • • •$_{n,}$ • • • • N$_0$
Persistent state: usrState • • U
Output: • • •true•false•
1: Extract nymble$^*$ from cred in usrEntries•sid• in
   usrState
2: return •nymble$^*$ •$\overset{?}{}$ blist•
```

Algorithm 7 NMSignBL
```
Input: •sid•t•w•target•blist• • • • N$^2$• • • • •$_{n,}$• • N$_0$
Persistent state: nmState • • N
Output: cert • •
1: Extract macKey$_N$•signKey$_N$ from keys in nmState
2: content •• sid••t••w••target••blist
3: mac •• MA•Mac•content•macKey$_N$•
4: sig •• Sig•Sign•content•signKey$_N$•
5: daisy •• target
6: return cert •• •t•daisy•t•mac•sig•
```

Algorithm 8 VerifyBL
```
Input: •sid•t•w•blist•cert• • • • N$^2$• • •$_n$• •,• • N$_0$
Output: • • •true•false•
1: •t$_d$•daisy•t$_s$•mac•sig• •• cert
2: if t$_d$ • t • t$_d$ • t$_s$ then
3:     return false
4: target •• • $^{(t_d• t_s)}$•daisy•
5: content •• sid••t$_s$••w••target••blist
6: return Sig•Verify•content•sig•verKey$_N$•
```

Algorithm 9 NMVerifyBL
```
Input: •sid•t•w•blist•cert• • • • N$^2$• • •$_n$• •,• • N$_0$
Persistent state: nmState • • N
Output: • • •true•false•
1-6: Same as lines 1–6 in VerifyBL
7: Extract macKey$_N$ from keys in nmState
8: return mac •$\overset{?}{}$ MA•Mac•content•macKey$_N$•
```

Type	Initiator	Responder	Link
Basic	–	Authenticated	Confidential
Auth	Authenticated	Authenticated	Confidential
Anon	Anonymous	Authenticated	Confidential

Fig. 6. Different types of channels utilized in Nymble.

Algorithm 10 NMInitState

```
Output: nmState • •N
1: macKeyNP •• Mac•KeyGen••
2: macKeyN •• Mac•KeyGen••
3: seedKeyN •• Mac•KeyGen••
4: •encKeyN•decKeyN• •• Enc•KeyGen••
5: •signKeyN•verKeyN• •• Sig•KeyGen••
6: keys •• •macKeyNP•macKeyN•seedKeyN•
7:          encKeyN•decKeyN•signKeyN•verKeyN•
8: nmEntries •• •
9: return nmState •• •keys•nmEntries•
```

Algorithm 11 NMRegisterServer

```
Input: •sid•t•w• • • N^2
Persistent state: nmState • •N
Output: svrState • •S
1: •keys•nmEntries• •• nmState
2: macKeyNS •• Mac•KeyGen••
3: daisyL • •R •
4: nmEntries' •• nmEntries•••sid•macKeyNS•daisyL•••
5: nmState •• •keys•nmEntries'•
6: target •• •(L• t+1)•daisyL•
7: blist •• •
8: cert •• NMSignBLnmState•sid•t•w•target•blist•
9: svrState •• •sid•macKeyNS•blist•cert•••••••••
10: return svrState
```

Algorithm 12 ServerLinkTicket

```
Input: ticket • •
Persistent state: svrState • •S
Output: • • •true•false•
1: Extract lnkng-tokens from svrState
2: •••nymble••••• •• ticket
3: for all • • • to •lnkng-tokens• do
4:     if •••nymble• • lnkng-tokens••• then
5:         return true
6: return false
```

Algorithm 13 NMComputeBLUpdate

```
Input: •sid•t•w•blist•cmplnt-tickets• • • • N^2• •n• • m
Persistent state: nmState • •N
Output: •blist'•cert'• • •m• •
1: •keys•nmEntries• •• nmState
2: ••macKeyN•seedKeyN•
   encKeyN•••signKeyN••   •• keys
3: for • • • to • do
4:     •••••ctxt•••• •• cmplnt-tickets•••
5:     nymble' ••seed •• Decrypt•ctxt•decKeyN•
6:     if nymble' • blist then
7:         blist'••• • R •
8:     else
9:         blist'••• •• nymble'
10: daisy'L • R •
11: target' •• •(L• t+1)•daisy'L•
12: cert' •• NMSignBL•sid•t•w•target'•blist••blist'•
13: Replace daisyL and tlastUpd in nmEntries•sid• in
    nmState with daisy'L and by t respectively
14: return •blist'•cert'•
```

Algorithm 14 NMComputeSeeds

```
Input: •t•blist•cmplnt-tickets• • N• •n• • m
Persistent state: nmState • •N
Output: seeds • • m
1: Extract decKeyN from keys in nmState
2: for all • • • to • do
3:     •t'•nymble•ctxt••••• •• cmplnt-tickets•••
4:     nymble' ••seed •• Enc•Decrypt•ctxt•decKeyN•
5:     if nymble' • blist then
6:         seeds••• • R •
7:     else
8:         seeds••• •• •(t• t')•seed•
9: return seeds
```

Periodic Update

At the end of each time period that is not the last of the current linkability window, each registered server updates its svrState by running (see Algorithm 7) ServerUpdateStatesvrState (), which prepares the linking-token-list for the new time period. Each entry is updated by evolving the seed andcomputing the corresponding nymble.

Each registered user sets ticketDisclosed in every usrEntry in usrState to **false**, signaling that the user hasnot disclosed any ticket in the new time period.

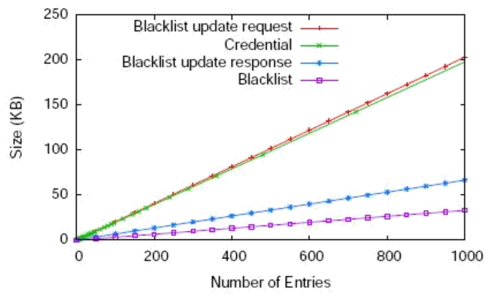

Fig. 7. The marshaled size of various Nymble data structures. The X-axis refers to the number of entries—complaints in the blacklist update request, tickets in the credential, tokens and seeds in the blacklist update response, and nymbles in the blacklist.

3.RESULT AND DISCUSSION

Figure 7 shows the size of the various data structures. The X-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 time periods 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 1 day with 5 minute time periods equates to L = 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 a complaint 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.

DISCUSSION

IP-address blocking By picking IP addresses as the resource for limiting the Sybil attack, our current implementation closely mimics IP-address blocking employed by Internet services. There are, however, some inherent limitations to using IP addresses as the scarce resource. If a user can obtain multiple addresses she can circumvent both nymble-based and regular IP-address blocking. Subnet-based blocking alleviates this problem, and while it is possible to modify our system to support subnet-based blocking, new privacy challenges emerges; a more thorough description is left for future work.

Other resources Users of anonymizing networks would be reluctant to use resources that directly reveal their identity (e.g., passports or a national PKI). Email addresses could provide more privacy, but provide weak blacklistability guarantees because users can easily create new email addresses. Other possible resources include client puzzles [25] and e-cash, where users are required to perform a certain amount of computation or pay money to acquire a credential. These approaches would limit the number of credentials obtained by a single individual by raising the cost of acquiring credentials.

Server-specific linkability windows An enhancement would be to provide support to vary T and L for different servers. As described, our system does not support varying linkability windows, but does support varying time periods. This is because the PM is not aware of the server the user wishes to connect to, yet it must issue pseudonyms specific to a linkability window. We do note that the use of resources such as client puzzles or e-cash would eliminate the need for a PM, and users could obtain Nymbles directly from the NM. In that case, server-specific linkability windows could be used.

Side-channel attacks While our current implementation does not fully protect against side-channel attacks, we mitigate the risks. We have implemented various algorithms in a way that their execution time leaks little information that cannot already be inferred from the algorithm’s output. Also, since a confidential channel does not hide the size of the communication, we have constructed the protocols so that each kind of protocol message is of the same size regardless of the identity or
current legitimacy of the user.

4. Future Scope

Our nymble project can be extended in next version called nymble and also can be developed on android platform. We are expecting 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. By providing a mechanism for server administrators to block anonymous misbehaving users, we hope to make the use of anonymizing networks such as Tor more acceptable for server administrators everywhere. All users remain anonymous misbehaving users can be blocked without deanonymization, and their activity prior to being blocked remain unlinkable . This work can also be extended into a multiple rounds of pseudonym construction in which the PM participates in multiple rounds of communication with the user. Another enhancement would be is to provide service providers with the ability to detect repeat offenders and revoke these users' access for longer durations of time.

5. 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 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.

6. REFERENCES

1. R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second Generation Onion Router," Proc. Usenix Security Symp. pp. 303-320, Aug. 2004.
2. Tor Project, available at www.torproject.org, accessed during June 2012.
3. Patrick P. Tsang, Apu Kapadia, and Sean W. Smith, "Nymble: Blocking Misbehaving Users in Anonymizing Networks" IEEE March-April 2011.
4. A. Lysyanskaya, R.L. Rivest, A. Sahai, and S. Wolf, "Pseudonym Systems," Proc. Conf. Selected Areas in Cryptography, Springer, pp. 184-199, 1999.
5. 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 Cryptographic Techniques (EUROCRYPT), Springer, pp. 93-118, 2001.
6. J. Camenisch and A. Lysyanskaya, "Signature Schemes and Anonymous Credentials from Bilinear Maps," Proc. Ann. Int'l Cryptology Conf. (CRYPTO), Springer, pp. 56-72, 2004.
7. 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.
8. D. Chaum and E. van Heyst, "Group Signatures," Proc. Int'l Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT), pp. 257-265, 1991.
9. D. Boneh and H. Shacham, "Group Signatures with Verifier-Local Revocation," Proc. ACM Conf. Computer and Comm. Security, pp. 168-177, 2004.
10. D. Chaum, "Showing Credentials without Identification Transferring Signatures between Unconditionally Unlinkable Pseudonyms," Proc. Int'l Conf. Cryptology (AUSCRYPT), Springer, pp. 246-264, 1990.
11. C. Cornelius, A. Kapadia, P.P. Tsang, and S.W. Smith, "Nymble: Blocking Misbehaving Users in Anonymizing Networks," Technical Report TR2008-637, Dartmouth College, Computer Science, Dec. 2008.
12. I2P2, available at , www.i2p2.de, accessed during June 2012.