CPG: Closed Pseudonymous Groups

Reed S. Abbott, Timothy W. van der Horst, and Kent E. Seamons
Internet Security Research Lab
Brigham Young University
Provo, Utah, USA
{rsa33, timv, seamons}@cs.byu.edu

ABSTRACT

This paper presents the design and implementation of Closed Pseudonymous Groups (CPG), a pseudonymous communication system for a closed user community (e.g., a class of students, team of employees, residents of a neighborhood). In CPG, each legitimate user is known by a pseudonym that, while unlinkable to a true identity, enables service providers to link users' behavior and blacklist any abuser of the system. This system is useful for providing honest feedback without fear of reprisals (e.g., instructor/course ratings, employee comments, community feedback for local politics). CPG is designed to be easy to understand, to implement (using existing techniques), and to use. This paper also presents the results of an initial user study that resulted in an important design change.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Authentication*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

General Terms

Security

Keywords

anonymity, anonymous feedback, privacy, usability

1. INTRODUCTION

Anonymity is advantageous, and even necessary, in circumstances where users provide negative or controversial feedback. Fear of embarrassment or reprisal can severely limit users' willingness to participate and speak freely with candor and forthrightness.

This paper presents the design and implementation of Closed Pseudonymous Groups (CPG), a system designed for pseudonymous communication amongst a closed group. CPG provides a pseudonym to each legitimate user in a group that cannot be linked to the user's identity. Service providers to the group are able to link a pseudonym's behavior and blacklist users that abuse the system.

The following are some examples of environments well-suited to CPG.

Student Feedback. A teacher desires candid feedback from students regarding all aspects of a course. The students do not want to suffer penalties or bias in how their work is graded. They also want to avoid negative repercussions from a teaching assistant if they provide a negative review of the TA's performance. The teacher wants to be able to distinguish between ten students reporting that the TA's performance is unacceptable versus one vocal student submitting ten reports that the TA is unacceptable.

Employee Feedback. A company wants to host an anonymous service where employees can offer suggestions or report on questionable practices. The employees want to avoid negative repercussions such as losing their job or negative bias in career advancement decisions. The employer wants a mechanism where employees feel safe to comment on important issues or situations that demand attention.

Parent Feedback. Parents want a forum to provide feedback to their local school system without fear of reprisal against their school-aged children. School personnel want to be able to block an offensive parent and to link feedback from each pseudonym so that they can detect important trends in the feedback. Parents who habitually complain about many aspects of the school may lose credibility unless there is corroboration with other's feedback.

Local Politics. A community would like a safe venue to freely discuss local political issues. Individuals may want to take unfavorable or controversial positions without alienating their neighbors or elected officials.

This paper presents the design and implementation of a system that serves the needs of these kinds of closed user communities. Even though many anonymous systems consider linkability a vice, CPG targets scenarios where it is helpful to link an anonymous user's actions in order to accurately interpret a user's activity over time. Misbehaving users can be blocked without revealing their identity or involving a trusted third party. The system was designed to be easy to understand, easy to implement using existing tech-

WPES'08, October 27, 2008, Alexandria, Virginia, USA.

niques, and easy to use. The results of an initial user study are reported. The user study led to an important design change to help make the system more foolproof.

The remainder of the paper is organized as follows: Section 2 discusses related work. Section 3 presents the design of CPG and Section 4 describes a prototype implementation. Section 5 contains a threat analysis and Section 6 reviews the key results from two usability studies. Section 7 contains conclusions and future work.

2. RELATED WORK

Network anonymizers, such as Tor (The Onion Router) [5] and JAP (Java Anonymous Proxy) [2], provide anonymity and unlinkability to clients that access network services. Tor obfuscates the origin of network traffic by allowing users to create and utilize an encrypted circuit through a series of anonymizing servers. Anonymity is achieved since each server only knows about the previous and next nodes in the circuit. In JAP, a client organizes a group of anonymizing proxies, known as a *mix*, into a *cascade* (a specific order for messages to pass through the proxies). All client traffic enters the cascade at the same point and travels through the same series of proxies. To a Web server, all traffic from a single mix appears to come from the same proxy. JAP attempts to defeat timing attacks (see Section 5.2) by generating filler data when a client is not active. CPG provides application-level anonymity and is therefore a complementary service to network-level anonymizers.

Nym [7] is a pseudonymous system for open systems such as Wikipedia. Users exchange a limited resource, like an IP address, for a blinded, signed token. The unblinded, signed token is then exchanged for a signature on a digital certificate from a certificate authority (CA). This digital certificate acts as a user's nym that can be blacklisted to limit the abuse of malicious users. The simplicity of Nym makes for a low barrier to adoption. The ease of obtaining email addresses and IP addresses may provide too little resistance to address the problem of malicious Wikipedia users. Nym and CPG both use Chaum's blind signature technique [3, 4]. CPG incorporates alternative authentication techniques that are well-suited to creating closed groups of the kind described in the motivating scenarios.

Resnick and Friedman [6] discuss some of the problems of being able to acquire cheap pseudonyms. They describe identifier commitments using blind signatures as being analogous to carbon paper and envelopes. A server signs the outside of an envelope that contains carbon paper such that the server's signature is affixed to the envelope's contents without the server learning the contents. Both Nym and CPG issue tokens using this basic approach.

Johnson et al. [8] present Nymble, a system where honest users remain anonymous and a server can blacklist a user without revealing the user's identity. Nymble supports unlinkable actions. The pseudonym manager and Nymble manager can collude to reveal a user's identity. In CPG, the managers cannot determine the identity of a pseudonym.

Tsang et al. [9] introduce a system that is able to blacklist users without knowing the identity of misbehaving users. It is designed for unlinkable pseudonyms actions. CPG also supports the ability to blacklist users without revealing their identity. It is designed for applications where linkability of a pseudonym's activities is desirable.

3. CPG DESIGN

Closed Pseudonymous Groups (CPG) is a framework for providing anonymity within a closed group of users. CPG has the following design goals:

- An administrator can restrict nym creation to a set of authorized users
- A client's true identity and nym are unlinkable
- A client's actions with a nym are linkable
- An administrator can block a misbehaving nym

3.1 CPG Framework

The CPG framework consists of three entities and four stages of operation. The entities are: 1) Membership server; 2) Service provider; and 3) Client. The membership server verifies that a client is a member of a group and enables that client to register a pseudonym with the service provider. As this system is resilient to collusion between the membership server and the service provider, both these entities may be hosted on the same system.

In Stage 1, the client authenticates to a membership server, which then signs a client-generated, blinded membership token. In Stage 2, the client waits before using the membership token to prevent timing attacks. In Stage 3, the client accesses a service provider to register a pseudonym with the service. The service validates the client's signed, unblinded membership token and authenticates the client's nym. In Stage 4, the client accesses the service provider using its established nym. Stages 1-3 are an initialization phase where a user establishes a nym with the service provider, and stage 4 can be repeated for the lifetime of the nym.

3.1.1 Stage 1: Membership Token Acquisition

In this stage, the membership server signs a single membership token for each authorized client. This process begins when clients submit and authenticate to their unique identifiers (e.g., email addresses, digital certificates, student ID numbers, corporate usernames). Authentication to an identifier varies according to the type of identifier. For example, with a digital certificate the client must prove ownership of the associated private key and with a corporate username a password-based authentication might be required. After a client is authenticated, the membership server checks to see if the client has already been issued a signed token.

If the client is permitted to receive a token, the client generates a large random value for the token. This token value should be large enough to avoid potential collisions with other clients. The client then blinds the token to prevent the membership server from linking a real identity to a nym when the token is later submitted to the service provider. After the blinded token is received and signed by the membership server, the signature is returned to the client, where it is unblinded. The following is a formal description of the blinded token generation based on Chaum's blind signature technique for digital cash [3, 4].

1. (Initial setup) The token server chooses a security parameter k and generates an RSA key with modulus n such that n is k bits long, a public exponent e and a private exponent d, then publishes $\{n, e\}$. It also chooses a cryptographic hash function:

$$h : \{0,1\}^* \rightarrow \{1..n-1\}$$

Stage 1 Membership Token Acquisition	**Stage 2** Wait	**Stage 3** Nym Registration	**Stage 4** Service Access
• Server authenticates client • Server signs blinded membership token (if one has not already been issued to this client) • Client unblinds token	...	• Server verifies unblinded token • Server authenticates nym • Server records token as used • Server authorizes nym for service access	• Server authenticates nym • Client uses service

Table 1: Summary of CPG stages

2. The client generates two random integers, a token $r \in \{0,1\}^*$ and blinding factor $b \in \{1..n-1\}$. It sends the server:

$$b^e h(r) \ (mod\ n)$$

3. The server decides whether to issue a token to the client. If a token is not issued, the protocol is terminated. Otherwise, the server signs the token and returns this value to the client:

$$(b^e h(r))^d \equiv b h(r)^d \ (mod\ n)$$

4. The client unblinds the signed, blinded token to obtain the signed token $t = h(r)^d \ (mod\ n)$ using the multiplicative inverse of b:

$$t = h(r)^d \equiv b^{-1} b h(r)^d \ (mod\ n)$$

Since the values $h(r)$ and $b^e h(r) \ (mod\ n)$ cannot be correlated due to the random choice of b, the transcript of the session cannot be used to identify the client when it later "spends" the token by revealing r and $h(r)$. To ensure that the server hasn't tried to "tag" the user with an invalid signature, the client verifies the signature to ensure that t really equals $h(r)^d \ (mod\ n)$:

$$t^e \equiv ?\ h(r) \ (mod\ n)$$

3.1.2 Stage 2: Post-Creation Delay

Once a client receives a signed membership token, she should wait a suitable time period before using it to access a service. If a client received a token from the membership server and immediately used the token to register her nym, a colluding membership server and service provider may be able to correlate the two actions based on timing. This attack is described in more detail in Section 5.2.

The appropriate waiting time depends on the CPG application. In a classroom setting, students may be given the first week of a semester to register a nym but are unable to use it. Thereafter, students may use their nym but no new nyms may be registered. In a corporate environment new clients may need to be brought into the system on an irregular basis. In this case, the length of the waiting period depends on the rate at which new clients are introduced. This is an area that must be carefully considered when CPG is incorporated into a new environment and is discussed in more detail in Section 3.3.1.

3.1.3 Stage 3: Nym Registration

In this stage, a client registers a nym with the service provider using the signed *unblinded* membership token. The client should use an anonymizer when communicating with the service provider (e.g., Tor [5], JAP [2]) to hide identifying information (e.g., IP address, browser type, cookies).

The submission of a valid membership token entitles a user to register a nym. The user must be able to demonstrate nym ownership. For example, a client might register local non-descript username (i.e., tk421) with the service provider and establish a password that will be used to prove nym ownership. Alternatively, a self-signed digital certificate or a third party identifier (e.g., an off-site email account) might also be used to register a nym.

Once the nym is registered, the client can discard the membership token and signature. The service provider keeps a record of the token to prevent the client from registering additional nyms.

3.1.4 Stage 4: Service Access

During the valid lifetime of their nyms, clients can authenticate and gain access to the service provider as a member of the pseudonymous group. All the actions of a particular nym are linkable. If one member is abusing their anonymity or otherwise misbehaving the administrator can disable or limit future access for the client using that nym.

3.1.5 Comparison to Digital Cash

CPG can be likened to a digital cash system where the membership service is a bank that issues one coin to each account holder, and user's can spend the coin once at the service provider to establish a pseudonym.

3.2 Nym Selection

A nym must be chosen carefully so that it does not reveal any identifying information about the client. For example, a client that is partial to cycling should avoid the nym `bike-boy`. Likewise, an employee should not use a work-based email as a nym to an external service since simply having a nym from the employer's domain reveals potentially identifying information about the client. An email address from a public email provider (e.g., Yahoo! Mail, Gmail, or Hotmail) may be less likely to reveal any information about the client and can be created anonymously. When communicating with an online service to create a nym, the client should use an anonymizer. For instance, when creating an email nym at Gmail, an anonymizer should be used to prevent Gmail from colluding with the service provider to determine the client's identity.

3.3 Adding and Removing Clients

Adding and removing clients must be handled appropriately to maintain client anonymity. In a service such as a classroom message board, suppose clients are added to the group at the beginning of the semester during a registration period before anyone accesses the service. The anonymity set is the total size of the group. This process is more problematic when clients must be added or removed on an irregular basis.

3.3.1 Adding Clients

When adding new clients to an existing group, a new client's anonymity is proportional to the total number of new clients added as a batch to the group (Stage 1) before they access the service (Stage 3). If a single client is added to a fully active group, it is trivial to link the new nym with the new individual that begins to participate in the group. If two new clients are added to an existing group, each new client is only slightly more anonymous within an anonymity set of size 2. The larger the number of new clients added together to an existing group, the greater the anonymity afforded each new client.

New clients may be added to a group as a batch or the entire group may all register for new nyms. With the batch method, deciding when to add new clients to a group depends on both the activity level of the service and the frequency with which new clients are added to the group. If activity level on a service is extremely low and there are many nyms that are unknown to the active users, it may be safe to add a new client to the group immediately; a batch of one. On the other hand, if activity level is extremely high it is unsafe to add a new client until a large group of new clients may be added. If the frequency with which new clients are ready to be added to the group is high, the waiting period for a new batch is low. A threshold method may be applied when deciding when to add new clients; for instance, no single client can be added until thirty new clients are ready to be added. It is important to remember that when a new client is added in a batch, the new client is only indistinguishable from the set of individuals with which they were added. The batch method is only reasonable if new clients are added on a regular basis.

If the frequency for adding new clients is low, the waiting period could be substantial. If this is the case, it may be best to invalidate all current nyms and force all clients to create a new nym. Although this method may be inconvenient, it maintains an anonymity set the size of the entire group.

3.3.2 Removing Clients

Removing specific group members from the pseudonymous group presents an interesting challenge as the identities of clients are not linked to their nyms. For example, if a student decides to drop a class, the professor should be able to prevent the student from being able to post to the class discussion board, however, the professor cannot easily do so as this student's true identity and nym are unlinkable.

As individual clients know their corresponding nyms, one approach is to require departing clients to reveal this mapping (e.g., demonstrate ownership of their nym) to the administrator. Once the link between the client's true identity and nym is known, it is trivial to remove authorization for that client. In addition to destroying any anonymity for the actions associated with this nym, with respect to the administrator, invalidating a single nym may enable others to link the client with their nym in a more subtle fashion. For example, if the disappearance of an extremely active member coincides with a student no longer coming to class, it is easy to infer the true identity of the nym.

A second approach is to invalidate the nyms of the entire pseudonymous group and require that all remaining members create new nyms for a new group. Barring distinguishing behavioral and contextual clues (e.g., impassioned championing of a particular cause, distinctive writing style, tendency to be very vocal on every issue), this approach prevents clients from linking a disappearing nym with a newly departed member of the group. This approach has the potential to be very inconvenient to users, especially if clients are removed from the group on a frequent basis.

A third approach offers a compromise between convenience and barring access to old group members. In this approach, the nyms of the entire group are invalidated on a periodic basis. The length of each period (e.g., semester, quarter) allows administrators to balance user convenience and the potential for abuse by former members. At worst, erstwhile members can continue to use the system until the end of the current period.

4. CPG IMPLEMENTATION

In order to study the usability of CPG, a proof-of-concept prototype of CPG was implemented. A membership server and service provider were created by extending the Phorum online message board (`http://phorum.org`) and, for simplicity, were hosted on the same machine. Group members are uniquely identified using their email addresses. Member also obtained separate email accounts to use as their nyms. Simple Authentication for the Web (SAW) [10] was used to authenticate ownership of these email addresses.

4.1 SAW: Simple Authentication for the Web

Simple Authentication for the Web (SAW) is a Web single sign-on method designed to address the problem of too many passwords for Internet users today. Sometimes it feels like every site on the Internet requires a username and password. As a result, users are faced with two options: they may either generate a new password for each site, which makes password management a difficult chore, or use a single password for all sites and trust each to guard the password. Neither option is ideal.

With SAW, the observation is made that in order to reset a password, users are often issued a temporary link, which is sent to the user's email account. Proving ownership of the email account is used as an authenticator. Since this method is already used on a regular basis and proven both usable and reasonably secure, why not use this method as a primary method of authentication? SAW does just that.

SAW works in the following manner. When a user wishes to authenticate to a service, the user simply enters an email address. Two tokens are issued to the user: one through the HTTPS connection established with the service, and one in an email to the user's email address. The user must be able to retrieve the token from their email account and resubmit both tokens to the content provider to successfully authenticate.

The process of checking the user's email for the token and resubmitting the two tokens can be automated through client side software. The user needs only a single password to

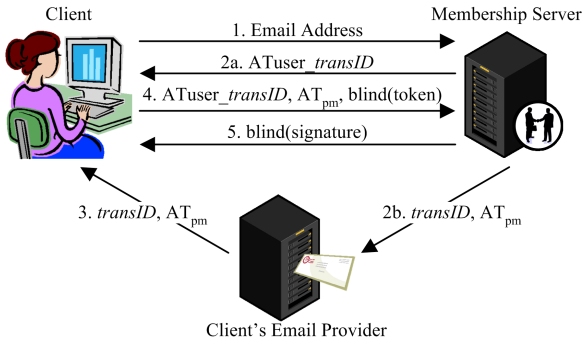

Figure 1: Membership token acquisition (Stage 1).

allow the software access to the email account. The software is then able to retrieve any tokens submitted to that email account and return them automatically. This creates a single sign-on solution that is simple to use, easy to implement, requires very little change of either the client or the server and no change to the email provider. SAW is impervious to eavesdropping and significantly raises the bar for active attacks.

4.1.1 Token Signing

To create a new group, the administrator specifies the email addresses of the group members to the CPG message board service. This enables the membership server to determine who is in the group. In order to enforce the requisite post-creation delay of Stage 2, the administrator also disables the login page to the service provider.

The client starts by navigating to the registration page and submitting their email address (see Figure 1). The server examines the authorization ACL for the client's address. If present, three large random numbers are generated and stored with the client's email address: a random transaction identifier and two SAW tokens. The first SAW token, AT_{user}, is returned directly to the user's browser as the value of a cookie named "ATuser_*transID*" (where *transID* is the transaction identifier).

The second token, AT_{pm}[1], and the transaction identifier are sent to the user's email account with a specially formatted subject line:

$$[\text{SAW-}URL]\ \text{transID}=transID\&\text{ATpm}=AT_{pm}$$

where *URL* is the address of the membership token signing page. This format is required by the SAW toolbar to facilitate identification of the appropriate email message. The body of the email is human readable, with instructions and a link to the signing page. The link has the transaction identifier and email token in the query string, similar to the email subject. When a client follows the link in the email, they are taken to the signing page, and the user token in the cookie is automatically delivered to the authentication server, while the email token and session identifier are delivered in the URL query string. These tokens are then stored as hidden input fields on the token signing page.

[1]In addition to email, SAW can leverage a variety of personal messaging mediums (e.g., instant and text messaging). As such, "*pm*", rather than "*email*" is used to describe this token.

If the email address is not on the ACL, a human-readable message is sent to the address explaining that an attempt was made to authenticate. The AT_{user} token is also returned to the client, even though it is not stored and the other parameters are not generated. Always returning this token prevents an attacker from probing the server to learn the contents of the ACL.

Once the client has reached the signing page, they are ready to generate and submit a membership token, $token_r$, for signing. JavaScript is used to generate a large random number for the membership token. Before the token is submitted to the membership server, the token must be blinded. Another random number is generated as the blinding factor used to obscure the token. The blinded token is then submitted to the membership server for signing along with the SAW tokens and transaction identifier.

Once the membership server receives the blinded token and SAW tokens, it first checks to make sure the SAW tokens are valid with the transaction identifier received. If the tokens are valid, the email address associated with the tokens is retrieved and the tokens removed from the database. The email address is removed from the ACL to prevent a client from receiving more than one signed token. Finally, the membership server signs the blinded token and returns it to the client.

The blinding factor must be stored by the client between submission to the membership server and the return of the blinded token. This implementation provides two viable alternatives. The first approach relies on AJAX (Asynchronous JavaScript and XML)[2]. Using AJAX, the tokens are submitted to the server and the signature received without refreshing the entire web page. The blinding factor remains stored in a JavaScript variable without any interaction from the user and without revealing it to the membership server. This approach is attractive because it requires no additional client-side software and side-steps JavaScript's inability to access the client's local file system to store persistent data. A drawback to this approach is that JavaScript is server-supplied code. This creates a "fox guarding the henhouse" problem as clients rely on the membership server to keep the blinding factor local to the client. The second approach, the CPG-enabled SAW toolbar (see Section 4.3), eliminates these concerns as the toolbar handles the token generation process.

4.2 Nym Registration

To begin, the client obtains a nym email account. Any email account can be used, but for this implementation users are advised to sign up for a webmail account and follow the guidelines of Section 3.2. Ideally the webmail account should provide POP or IMAP service and deliver email quickly to ensure reasonable login times. Some webmail services require a user to provide their primary email account when creating a new account. These services should be avoided since it provides a link between a client and their nym. Based on these criteria, two webmail providers are recommended: GMail.com and Gawab.com. GMail provides free, and consistent, POP and IMAP access over TLS. Authentication using the SAW toolbar takes about five seconds. Gawab.com also provides free POP access to their webmail accounts. The email delivery times are variable; sometimes authentication takes less than three seconds and other times

[2]http://www.w3.org/TR/XMLHttpRequest/

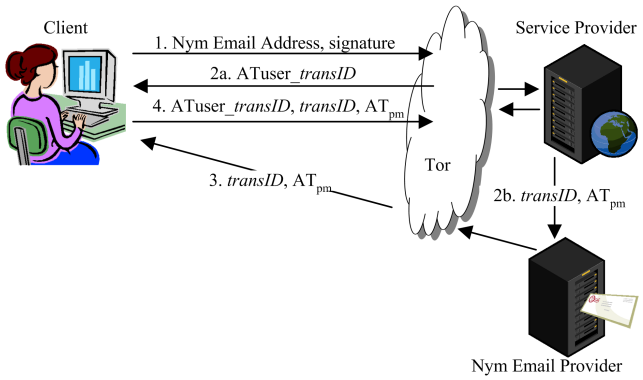

Figure 2: Nym registration (Stage 3).

it take almost thirty. Hotmail and Yahoo! Mail also provide POP service, but users are required to pay, so these were not examined for their usability with SAW.

The client can now register their nym (Figure 2). The client begins by submitting their nym to the service provider. Once again the client must authenticate using SAW, but this time using the nym email account. An access control list is checked to see if the nym is blocked. If not, the SAW tokens are created and distributed. After authenticating to the nym account, the server checks to see if the nym is already known. If the nym is already known to the system, the user is logged in. If the nym is unknown, the client must submit the unblinded membership token received from the membership server. Once received, the service provider verifies the signature and, if valid, the client is logged in and the nym and the token are stored at the service provider.

4.3 Toolbar

In order to make CPG more usable and secure, the SAW toolbar for Firefox has been extended to aid users in acquiring a signed token and using a nym. The CPG-enabled SAW toolbar is able to distinguish whether a page is requesting a nym or a standard SAW email account. The toolbar also manages signed tokens for the client and warns the user of potential mistakes. Hereafter, the CPG-enabled SAW toolbar is referred to as the CPG toolbar.

When configuring an account, the CPG toolbar allows a client to distinguish the nym accounts from a non-nym account. An additional 'nym' checkbox has been added to the account configuration screen. By checking this box, the client allows the toolbar to know which accounts should only be used as nyms. This helps prevent a client from accidentally submitting an account that can be tied to the client.

In order for the CPG toolbar to recognize which type of account is being requested, the HTML page requested must contain a form with a special name. To recognize that a page is SAW enabled, and thus requesting a standard account, an HTML form must exist with the name `saw_login_form`. CPG extends upon this idea. To recognize if a page is requesting membership tokens for signing by the membership server a form must exist with the name `cpg_register_form`. If this form exists the CPG toolbar enables a button. When a user clicks this button, the CPG toolbar is able to do the requisite blinding and submission of tokens. The signature received in response is unblinded and stored with the token value and domain that signed it.

As with registration, if a page is requesting a nym for logging in, a form must exist with the name `cpg_login_form`. When a client attempts to submit a nym, the toolbar checks to see if Tor is running. If it is not, the client is given a warning and the chance to cancel the submission. If the nym is submitted, the signed token from the domain is submitted as well. Although not required with every log in, the signed token is submitted by the toolbar whenever a client logs in. The token can be deleted after the first successful log in, but it is difficult for the toolbar to know if the login attempt is successful, and it is harmless to submit the token with every attempt. The client is logged in if they are able to authenticate to the nym email address and the signature on the token is valid.

As mentioned previously, if the CPG toolbar is not used, JavaScript, provided by the membership server, is used to generate and blind the membership token. This gives the membership server the opportunity to act maliciously. The membership server can easily generate a token it knows how to unblind or many other methods of leaving the option to link a client's nym and true identity. The client has the option of examining the code, but this is laborious and unreasonable. Without the toolbar, the client is forced to trust the membership server. For this reason, the toolbar makes CPG both more usable and secure.

4.4 Anonymizers

CPG relies on an anonymizer to obscure a client's IP address and scrub other information, such as a client's browser type, when the client accesses the service provider. Three different anonymizers were analyzed and tested for use with CPG: Tor, JAP, and anonymizing websites. Their strengths and weaknesses are summarized in Table 2.

4.4.1 Tor

Tor (The Onion Router) has several attributes that make it an excellent anonymizer. First, Tor uses a circuit of encrypted links to obscure a client's IP address. The requested web service only sees the IP address of the last computer in the circuit, not the IP address of the client requesting information. Second, Tor leverages the scrubbing mechanisms of Privoxy to cleanse requests of identifying information such as a client's browser type and persistent cookies. Third, to make Tor more user-friendly, a Firefox extension allows for quickly enabling and disabling Tor. Finally, Tor makes use of a series of anonymizing proxies instead of a single proxy. This prevents a user from having to place a large amount of trust in a single source. These qualities provide a high level of anonymity and make Tor moderately easy to use.

Tor has several drawbacks. First, Tor's throughput is highly variable. Performance varies depending on the number of active Tor users, the quality of Tor servers, and many other factors. Some informal tests we performed showed that it is not uncommon to wait over twenty seconds for a page of 18 kilobytes to finish downloading. Without Tor this same page normally takes less than three seconds to download. This corresponds with previous findings [5]. Second, Tor is not an option if a client is unable to install software (e.g., a public library computer, a corporate computer, a campus computer). Finally, it can be unclear whether or not the browser is routing through Tor. In our usability tests, one user assumed the browser was using Tor when first installed, when in fact it was not. The Tor icon appeared in the sys-

Anonymizer	Strengths	Weaknesses
Tor	• Fast (once the initial circuit is created) • Easy to setup and use with Firefox • Trust for anonymity not placed on a single source	• Initial circuit construction may be slow • Requires client-side software • May be unclear if browser is configured to use Tor • Inconsistent quality of service
JAP	• Easy installation • Trust for anonymity not placed on a single source	• Manual browser-configuration • Requires client-side software • Greater anonymity means slower transfer speeds
Website Proxy	• No software installation • Very easy to use	• Trust for anonymity placed on a single source • Non-TLS enabled proxies may add additional vulnerabilities

Table 2: Summary of the strengths and weaknesses of different anonymizers.

tem tray so the user assumed that traffic was being routed through the Tor network.

4.4.2 JAP

JAP (Java Anonymous Proxy) also has several positive traits. A positive aspect of JAP is that, like Tor, a client's trust is distributed over several anonymizing servers. No single server could be compromised to reveal a client's actions. JAP is also capable of scrubbing browser and operating system information from HTTP headers.

Performance of JAP varies. The throughput of JAP depends on the number of clients using the cascade. A larger number of users provides greater anonymity but reduces performance. Although a user must set the browser's proxy settings manually to use JAP, the JAP user interface provides a button that easily enables and disables its anonymizing capabilities.

As with Tor, the JAP interface can also be unclear about whether or not traffic is being routed through the service. A button on the JAP interface indicates if JAP is on or off, but does not indicate if a client's browser settings are correct. A meter on the interface displays the amount of activity a client is generating, but the meter does not reflect any activity until after the client transmits some traffic, at which point the user may have already inadvertently revealed their IP address to the service provider.

4.4.3 Anonymizing Websites

Anonymizing websites provide an easy way for clients to browse a target website anonymously. The anonymizing site then acts as a proxy between the target site and the client. This is accomplished by rewriting any cookies or links returned from the target site to refer to the anonymizing website, thus forcing the client's traffic to the target site to flow through the anonymizing site. The anonymizing website must be allowed to access the target site.

The primary advantage of an anonymizing website is that no client-side software needs to be installed. This means that a client can attain some anonymity from any computer, provided the anonymizing website is not blocked by web filtering software. Many anonymizing web websites also offer to block cookies, scrub identifying information, and block dynamic content. The other advantage of the anonymizing website is that it is extremely easy to use. There is no configuring browser settings or fiddling with buttons to turn it on and off.

The disadvantage of an anonymizing website is that a great deal of trust is placed in a single source. Logs maintained by the anonymizing site can be used to trivially link a client's true IP address with their actions. The anonymizing website is trusted to either not keep any logs or never reveal them. Perhaps the greater danger is that the anonymizing website is acting as a trusted man-in-the-middle. Since the anonymizing website is able to see all data flowing between the client and the destination website, the anonymizing website is trusted not to abuse this information. Such information could include usernames, passwords, session cookies, credit card numbers, social security numbers, etc. In some cases, these sites can reduce security. Many do not offer a TLS-enabled connection. If a client is visiting a site that is protected by TLS, then by rerouting through the anonymizing site that does not offer TLS, all data between the client and anonymizing website is unencrypted. This leaves the information vulnerable to an eavesdropper between the client and the website.

5. THREAT ANALYSIS

The major threats to CPG are weaknesses in the chosen authentication mechanism, timing attacks, user tracking, and user error. This section discusses these threats and supplies recommendations to avoid or mitigate them.

5.1 Authentication Mechanism

Since CPG can be implemented with various authentication technologies, an implementation inherits the threat model of the chosen authentication technology. Our prototype implementation created using SAW is susceptible to the threats and risks that SAW introduces [10]. For example, the basic SAW protocol is susceptible to an active attack since SMTP traffic is often not secured. If an attacker knows an authorized pseudonym email address associated with a nym, an active attack works as follows during Stage 4. The attacker submits the pseudonym email address to the service provider. The server responds by generating the requisite tokens that are used to authenticate the client. The user token is delivered directly to the attacker and the second token is emailed to the pseudonym email address. If the attacker is able to eavesdrop on the communication between the service provider and the email provider, she will be able to intercept or observe the email token as it is being delivered. Since the attacker has now acquired both tokens, she is able to authenticate as the victim and access the service.

This attack is also feasible against the registration process, however, it must occur before the proper owner receives a token. In this case, the real owner will detect the potential abuse when the system reports that the membership token has previously been issued.

This attack can also occur during Stage 1 if an attacker impersonates a legitimate user to the membership server. However, since the server only hands out a single membership token to each authorized user, the attack must occur before the legitimate user attempts to acquire a membership token because the authorized user can detect the potential abuse when the system reports that the membership token has already been issued to this user.

These examples illustrates why the authentication method chosen for an implementation must have an appropriate risk level. SAW is appropriate for low to medium security situations. For more secure applications, digital certificates or two-factor authentication methods may be more suitable.

5.2 Timing

CPG is vulnerable to several timing attacks. First, even though a pseudonym is unlinkable to its user's true identity, a membership server and a service provider could collude and compare access times in order to infer the identity of a pseudonym. If a user interacts with the service provider immediately after obtaining a signed membership token from the membership server, such behavior could leak his true identity.

Second, suppose a service provider has access to a client's local network and is able to track network activity. If the client contacts an anonymizer, such as Tor, just prior to the anonymizer accessing the service provider, then the service provider can correlate these actions and reliably infer the pseudonym's true identity. This timing attack could occur when the user's employer hosts the anonymous service and the user accesses the service from a computer at work.

***Recommendation** 1.* Once the user acquires a signed token, she should delay contacting the service to reduce the risk of a timing attack. One way to address this issue is for the system to provide a distinct membership registration period that closes before access to the anonymous service is allowed. Without a designated registration period, a client must be able to hide her activity among others who are accessing the system. This may work best when the registration system and anonymous service are not hosted by the same organization, and the users can be given suitable, trustworthy guidelines for how long to delay accessing the system after they have registered.

***Recommendation** 2.* A client should access services from a network that is not controlled or accessible to the organization hosting the service in order to avoid the second timing attack.

5.3 User Tracking

The membership server and the service provider can collude, possibly with a third party, to track a user's behavior and discover the true identity of a pseudonym. Using cookies, the membership server may be able to cause the user's computer to contact the membership server and reveal the user's identity when the user accesses the anonymous service.

For example, suppose a membership server in the same domain as the service provider, or a third party colluding with the membership server and the service provider, places a cookie on the client's machine when the client registers. The cookie can contain information used to track and identify the client: the client's email address, the time the token was signed, or a tracking number associated with that client. When the client contacts the anonymous service, a web page containing a web bug that references the membership server or the third party will result in submission of the cookie that reveals the identity of the nym owner.

***Recommendation** 3.* To prevent this attack, the client should refuse all third-party cookies and destroy all cookies from the membership server's domain after having a membership token signed. All web browsers allow a client to delete cookies manually. In some browsers (e.g., Firefox) cookies can be destroyed automatically at the end of every session. This setting is recommended if it is available. Turning cookies off completely thwarts this attack, but many websites require cookies for functionality.

***Recommendation** 4.* Web browsers should provide mechanisms to easily specify when third party cookies should be disabled, and also support a restricted mode where all content is retrieved and submitted to a single domain. Service providers offering anonymity should concentrate all of their content on a single site to assist users in safe practices.

5.4 User Error

Users are often the weakest link in the security chain, and there are several opportunities in CPG for user's to sacrifice their anonymity. First, users may inadvertently disclose their identity by the input they provide, especially when that input is free-form text. Certain phrases or writing style could leak the author's identity. Second, a user could mistakenly bypass the network anonymization system like Tor and contact a service directly. Our experience shows that users are often unclear about whether their network traffic is actually flowing through an anonymizing network. Mistakes can leak IP address information and other data used to identify to the user. Third, a user could make a mistake during authentication with the service and reveal the wrong username/password, email address, certificate, or other authentication tokens that reveal identity.

***Recommendation** 5.* Anonymizing services should provide structured input forms whenever possible so that user input is uniform and less likely to reveal identity information. For instance, a teacher could conduct a multiple choice survey about a course rather than ask the users to provide free form feedback.

***Recommendation** 6.* Network anonymizers need better visual cues to the user to indicate when they are active. A user may have installed a tool and believe it is active when it is not.

***Recommendation** 7.* Anonymization tools should provide hooks for higher level applications to automatically invoke them or check whether they are active so that applications like CPG can automatically prevent inadvertent user errors. Another option is to have the underlying anonymization feature tightly integrated with CPG so that the service provider can only be contacted anonymously.

***Recommendation** 8.* Client-side filtering should include rule checking to prevent the inadvertent disclosure of the wrong authentication token to an anonymous service.

6. USABILITY

Two small usability studies were conducted with students in two undergraduate computer security courses at BYU. The students volunteered to participate in these studies, each of which lasted two weeks. Participants were directed to an online set of instructions detailing how to configure and use CPG. They completed the instructions at their leisure and on their own time. The students' identifiers were their email address associated with their registration in the course. They were given the first week to obtain a signed membership token and the following week they could sign in anonymously with their nym and post to the class message board. After the two week period was over, students were given a set of questions about their experience using CPG. A summary of the results is available [1].

The first study involved 5 students. A significant flaw in the original CPG design was discovered that led to the design presented in this paper. In the original design, a client's nym email address was hashed, blinded, and signed by the membership server instead of a membership token (see Section 3.1.1). This required students to create a nym prior to having it signed. Later, in Stage 3 (see Section 3.1.3), clients were authenticated to the nym and required to submit the signature on their nym when logging in for the first time. This was problematic for two users. The first user, instead of having their nym email account signed, made up an unrelated, non-email nym and had it signed instead. In Stage 3, the user was unable to authenticate using the nym since it was not an email address. The second user mistyped their nym email address when configuring the nym account in the CPG toolbar. The user's misspelled nym was then submitted to the membership server and signed. When the user went to log in for the first time, the incorrectly spelled email was not on the ACL so an error message was sent to that address, which the user could not access. After realizing the mistake, the user attempted to have the correctly spelled nym signed, but their email address had been removed from the membership server's ACL to prevent them from having another nym signed.

The lessons learned in the usability study resulted in improvements to the CPG design to its current form. Clients had problems with signing their nym because they were not required to prove ownership before having it signed. The membership server was unable to examine the nym to verify its validity since it had been blinded. As a result, these users had the wrong nym signed. They were then incapable of authenticating to the nym and were unable to have the correct nym signed since their email address had been removed from the ACL. By signing a membership token instead of the nym, the client must authenticate to the nym before spending the membership token.

The second study involved 10 students. The introduction of the membership token made the system more foolproof, and no user experienced any difficulty registering a nym. The most significant feedback obtained was that the instructions could be improved. Also, most users found Tor inconvenient to install and operate. Two users even disabled it and proceeded without it, jeopardizing their anonymity. Finally, the SAW authentication mechanism works well when an email service has low latency, but is not desirable when email delivery is routinely delayed for more than a few seconds. A few users had email providers with high latency, which left them with a negative impression about the system.

7. CONCLUSIONS AND FUTURE WORK

Closed Pseudonymous Groups (CPG) is a novel framework for providing practical anonymity within a closed group of users. Clients can act anonymously, but administrators can block nyms that act maliciously. It is designed using existing techniques, is easy to understand, and has been shown to be easy to use by technical users.

CPG provides a strong separation between a client's nym and their true identity. This prevents outside sources from strong-arming an administrator into revealing a client's identity as, if it is used correctly, even the administrator is unable to discover a nym's true identity. This paper describes the common threats to CPG and provides a set of recommendations to mitigate those threats.

CPG provides mechanisms to remain anonymous, but it cannot prevent a user from indirectly disclosing their identity through misbehavior. A student who complains relentlessly about the same topic in class as well as on a CPG-enabled class message board leaves little doubt about their identity. The following is a list of assumptions for maintaining a user's anonymity and can serve as a checklist to a client when using CPG.

- A pseudonym should not reveal information about its owner
- A client computer should be on a separate network from the service provider
- A client should wait a sufficient time between receiving and using a membership token
- Cookies from the membership server should be deleted after receiving a membership token
- An anonymizer should be activated and configured correctly when interacting with the service provider
- A client should not reveal identity information to the service provider

CPG can be incorporated into an existing service with only minor changes, as demonstrated by our prototype application that integrates CPG into a discussion forum. CPG is designed to leverage existing authentication and authorization infrastructures within an organization.

The usability of CPG can still be improved. The process of installing an anonymizer and toolbar, configuring the toolbar, signing up for a separate email account, and then following one type-written page of instructions remains too tedious for most users. To simplify the process, the toolbar could be bundled with the anonymizer and installed concurrently. Using the toolbar, it might also simplify registration by authenticating and requesting a signed token in a single step. The user will configure the toolbar with their email address and navigate to the registration page. The toolbar will then perform SAW authentication and, if the authentication succeeds, the toolbar will automatically generate and have the membership token signed as well.

The usability of the current CPG prototype is dependent on the usability of SAW. If the email provider has high latency, then response times can frustrate users. Replacing email with instant messaging can reduce latency during authentication.

Future studies could observe the system in production use to determine whether user's behavior leaks their identity. An area for improvement is an enrollment/revocation process for a dynamic user base that balances privacy and ease of use.

Our usability study shows that it can be unclear to users whether or not traffic is being routed through a network anonymizer. Software that provides user-level anonymity should be able to verify that its assumptions of network-level anonymity are met. This would ensure fewer inadvertent disclosures of a user's true identity or other potentially identifying information. Potentially, user- and network-level anonymizers could be tightly coupled into a single application. Alternatively, network-level anonymizers (e.g., Tor, JAP) could provide an API that external applications could invoke to verify proper installation and usage.

8. ACKNOWLEDGEMENTS

This research was supported by funding from the National Science Foundation under grant no. CCR-0325951, prime cooperative agreement no. IIS-0331707, and The Regents of the University of California.

9. REFERENCES

[1] R. Abbott. CPG: Closed pseudonymous groups. Master's thesis, Brigham Young University, Mar. 2008.

[2] J. Anonymity & Privacy. JAP Anonymity & Privacy, 2006.

[3] D. Chaum. Blind Signatures for Untraceable Payments. In *Advances in Cryptology: Proceedings of CRYPTO '82*, Plemum, New York, 1983.

[4] D. Chaum. Security without identification: transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[5] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the Thirteenth USENIX Security Symposium*, San Diego, CA, Aug. 2004.

[6] E. J. Friedman and P. Resnick. The social cost of cheap pseudonyms. *Journal of Economics & Management Strategy*, 10(2):173–199, Summer 2001.

[7] J. E. Holt and K. E. Seamons. Nym: Practical Pseudonymity for Anonymous Networks. Technical Report 2006-4, Brigham Young University, June 2006. `http://isrl.cs.byu.edu/pubs/isrl-techreport-2006-4.pdf`.

[8] P. C. Johnson, A. Kapadia, P. P. Tsang, and S. W. Smith. Nymble: Anonymous ip-address blocking. In *Privacy Enhancing Technologies Symposium*, Ottawa, Canada, June 2007.

[9] P. P. Tsang, M. H. Au, A. Kapadia, and S. W. Smith. Blacklistable anonymous credentials: blocking misbehaving users without ttps. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 72–81, New York, NY, USA, 2007. ACM.

[10] T. W. van der Horst and K. E. Seamons. Simple Authentication for the Web. In *3rd International Conference on Security and Privacy in Communication Networks*, Nice, France, Sept. 2007.