SecurePost - An Android Tool for Trusted Social Media Posts

Pritha Doddahosahally Narayanappa

Abstract— **This project is part of the FlowNet initiative. FlowNet aims at providing Internet freedom and free flow information through socially informed, censor resistant online social networks. My contribution for FLowNet is in developing an Android application, SecurePost. The requirement for SecurePost is two-fold. First, the system should facilitate secure, anonymous, group communication within a closed group of trusted members. Second, the general public on the Internet viewing this content, should be able to verify that the content was generated only by the said closed group of trusted members. The system consists of an Android client application, a proxy server and a browser-plugin. The OSNs supported by this system are Twitter and Facebook.**

I. INTRODUCTION

The ultimate goal of this project is to promote democracy and protect human rights by allowing users to communicate freely, without interference from government and censorship agencies. Online social networks (OSNs) have revolutionized communication worldwide. The statistics are astounding. As of 2012, 1.2 billion people use Facebook, Twitter adds about 500,000 new users a day. 800 million unique visits to YouTube occur every month. Earlier work demonstrated that even residents of the far reaches of rural Africa use Facebook frequently to communicate with others in their community as well as with friends, family, and colleagues worldwide. In these areas, online social networks (OSNs) are critical as alternate communication infrastructures (e.g. fixed telephone lines, cellular service) are infrequently available. Yet OSNs have become invaluable not only for staying in contact with friends and family. The 2008 US Presidential election witnessed the most widespread use of social media in US elections to date. The Egyptian government understood the power of OSNs well enough to shut down nationwide Internet service in January 2011. Also in January 2011, protestors outside the Saudi Embassy in Washington displayed a banner thanking Facebook founder Mark Zuckerberg. These events and more have made clear that OSNs are an invaluable and irreplaceable communication tool worldwide. However, what is also clear is that access to OSNs is not uniformly available to the worlds citizens. There are two major impediments to access: government restrictions and censorship attempts, and lack of or limited Internet penetration in developing communities. Like Secretary of State Clinton stated in her February 15, 2011 address to GWU, we too feel it is critical to address the challenge of achieving both liberty and security of social media communication. We propose a highly transformative 3-year project that will create a set of solutions, called FlowNet, in the form of a censorship-resistant distributed social network. Our team will develop technical solutions that redesign OSN interaction and participation by addressing the accessibility, censorship, and privacy challenges that restrict OSN usage. Our interdisciplinary partnership will ensure that our technological solutions will be informed by a deep understanding of the user base so that we design solutions that are meaningful, usable and successful. We will infuse a humanist and social science component into our research to better understand and characterize OSN perceptions, usage, and needs by residents of our targeted communities so that the relevance and sustainability of our technical solutions is maximized. Our team has extensive experience in the development of wireless communication and OSN technology, including anonymity solutions, as well as social survey and ethnographic fieldwork. As we increase our understanding of user concerns about OSN safety, we can use this knowledge to fine-tune our technical solutions to ensure usability and applicability in our target communities. Further, because our team integrates the program evaluators, at each step we can assess impact and evaluate progress in meeting our stated objectives. Where possible, our technical solutions are based in part on well-known, well-tested, and popular technologies. Instead of building and bootstrapping our own platform with users, our approach extends and enhances access to existing OSNs, such as Twitter and Facebook. This leverages existing user populations, simplifies adoption and maximizes sustainability. Further, we base our technical solutions on handheld devices, including smartphones, feature phones, and

II. BACKGROUND

Systems for anonymous communication on the Internet, or anonymity systems, provide a technical means to enhance user privacy by hiding the link between the user and her remote communicating parties (such as websites that the user visits). Popular anonymity systems include Anonymizer.com [9], AN.ON [4, 16], and Tor [14]. Tor is used by hundreds of thousands of users, including journalists, dissidents, whistle-blowers, law enforcement, and government embassies. Anonymity systems forward user traffic through a path (or circuit) of proxy servers. In some systems, including Tor, the proxies on the circuit are selected from among a large number of available proxies, each of which is supposed to be operated by a different person. An attacker, however, might run a substantial fraction of the proxies under different identities. He would then be able to deanonymize users whose circuits run through his attacker controlled proxies. Thus, the security of the anonymity system hinges on at least some of the proxies in the circuit being honest. Having some means to discern which proxies are likely to be honest would thereby greatly enhance the security of the system.

III. SOLUTION

The intent is to achieve anonymous, verified group communication. The SecurePost Android Application is an interface to Twitter and Facebook that allows groups to post securely and anonymously. The SecurePost Verifier Browser Plugin, allows anyone on the Internet to verify the authenticity of a post.

A. Generating Cryptographically Signed Content

The content that is posted from the application needs to be verifiable for authenticity and integrity. In order to fulfill this requirement, a time-based one-way hash chain of keys is used to sign every post made by the SecurePost application. This idea is inspired from [1], which proposes a technique for computation of consecutive preimages of hash chains, given a seed. Consider a hash chain with a starting point v_0 and an endpoint v_n, where the latter is the seed from which the chain is computed. Each element v_i is the hash image of the next value on the chain, i.e., v_i = h(v_i+1). Our aim is to compute and output the series v_1, v_2, . . . , v_n in that order and in a manner that requires minimal memory and computational requirements. It is clear that previously output values are not useful in computing the next value to be output, since the hash function is one-way. Instead, the output values have to be computed by iterative application of the hash one-way function to a value towards the end of the chain. At any time, the current pointer (which corresponds to what element gets output) is in one interval of size 4; one of size 8; one of size 16; and so on, up to the length of the hash chain. A pebble is associated with each such interval, and ”strives towards the midpoint of the interval. Given the way the intervals are arranged, such midpoints constitute endpoints for smaller intervals. When the current pointer reaches a pebble, a new interval of the same size is created, adjacent to the old interval, and the pebble is started off at its end. Since this is a midpoint for a larger interval, another pebble will be found there. The newly moved pebble then moves to the middle of its own interval, taking a few steps for every output value we generate. With each pebble we associate a value, corresponding to the hash chain value at the location of the pebble. Thus, when a pebble is reassigned to a new interval, it obtains the value of the pebble in its aqcuired position; for each step it moves, it applies the hash function to its value. Given that the current pointer is always in intervals populated by pebbles, we can bound the computational effort to derive the output value from the pebble values.

B. Enabling shared group communication

The application needs to enable a selected group of users to share a Twitter or Facebook account, without sharing passwords. The application requires privileges to post as the user. The solution here is to use OAuth authentication. OAuth is an authorization framework that enables applications to obtain limited access to user accounts on an HTTP service, such as Facebook, GitHub, and DigitalOcean. It works by delegating user authentication to the service that hosts the user account, and authorizing third-party applications to access the user account.

An access token is an opaque string that identifies a user, app, or page and can be used by the app to make graph API calls. Access tokens are obtained via a number of methods, each of which are covered later in this document. The token includes information about when the token will expire and which app generated the token. Because of privacy checks, the majority of API calls on Facebook need to include an access token.

Twitter provides the 3−legged OAuth flow which allows the application to obtain an access token and access token secret by redirecting a user to Twitter and having them authorize the application. Facebook Page Access Tokens are similar to user access tokens, except that they provide permission to APIs that read, write or modify the data belonging to a Facebook Page. To obtain a page access token you need to start by obtaining a user access token and asking for the manage−pages permission.

C. Proxy Server

Secure, controlled and safe social media communication being the purpose of the application, distributing OAuth tokens to all members of a group is not a viable solution. This necessity can be met by introducing a reliable centralized Proxy server to store the OAuth tokens for each account. The users on the Android client relay all information exchange via the proxy server. The proxy server uses the stored OAuth tokens to post data to Twitter or Facebook. This design also has the advantage of never exposing individual user IP addresses to Twitter and Facebook. It adds added user identity protection.

D. Notifications using Google Cloud Messaging

When high impact information or information requiring timely comprehension and timely response, are shared on the shared accounts, it is required that all users are notified, though they may not have the application open at the moment. For the requirement to have all members of a group notified of a new tweet/post on the shared group, Google Cloud Messaging infrastructure and APIs are used.

The proxy server also acts as the Cloud Connection Server(CCS) to relay notification messages to all registered devices via the Google Cloud Messaging infrastructure. SecurePost uses the XMPP connection server protocol. XMPP (Extensible Messaging and Presence Protocol) is a protocol based on Extensible Markup Language (XML) and intended for instant messaging (IM) and online presence detection. CCS (XMPP) is asynchronous and persistent, which makes it faster than HTTP. In result, it helps keep battery and data usage to a minimum.

IV. DESIGN AND IMPLEMENTATION

This section describes the detailed design and implementation of the SecurePost system, discussing the security and work flow of the application. First, the process of registering the account and users with the proxy server is described. Second, the process of bootstrapping on the phone is described.

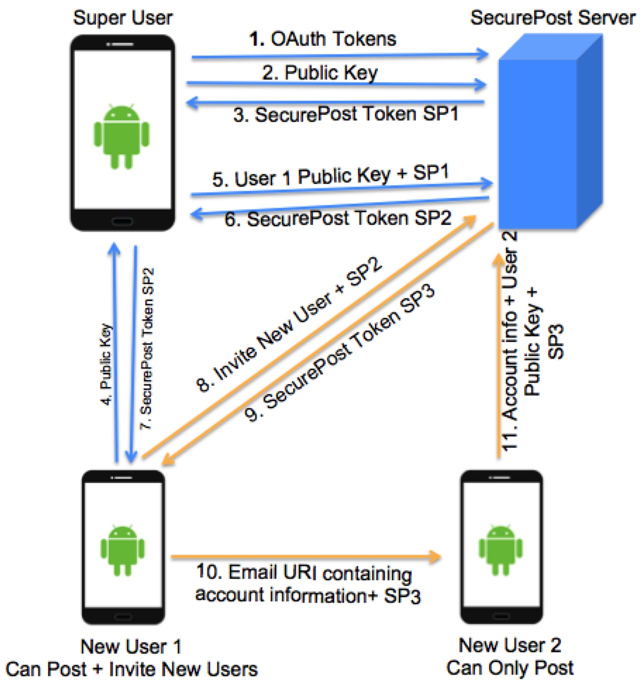

Fig. 1. Client Server Registration Process

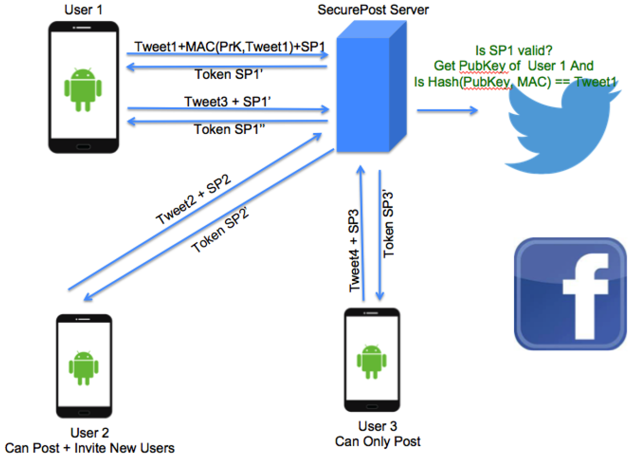

Fig. 2. The process of tweeting or Facebook posting via Proxy Server

This step involves generation of the hash chain, comprising only of the pebble values, as described earlier. Third, the server implementation of the steps involved in posting social media content is discussed. Lastly, the IOCipher encrypted file system is discussed.

A. Registering with Proxy Server

Figure 1. depicts the process of setting up a shared account as well as registering members on the server.

- Step 1 is for the Super User to login to Twitter/Facebook account that is to be shared and authorize SecurePost to post as the user/ as the page. An RSA public/private key pair is generated on the phone. All information uploaded to the proxy server, is signed with this private key. The Super User sends the public key along with OAuth tokens. Step 3 shows the server returning a unique token, which the proxy associates with every member of a group.
- Steps 4-7 show how a new member can be added to a group, in an in-person scenario. The new user presents his/her public key stored as a QR code to the Super User, who then uploads it to the proxy server. The Super User in turn relays the unique token the proxy associated to the new user.
- Steps 8-11 show how a new member can be added to a group, over the Internet. The member who is inviting a new user, informs the proxy of this intention. He relays a token along with account bootstrap information to the new user over Email. The application has an IntentFilter set, and opens the SecurePost application if installed. In cases where the application is not already installed, the user is redirected to http://securepost.info/, the official webpage of the application.

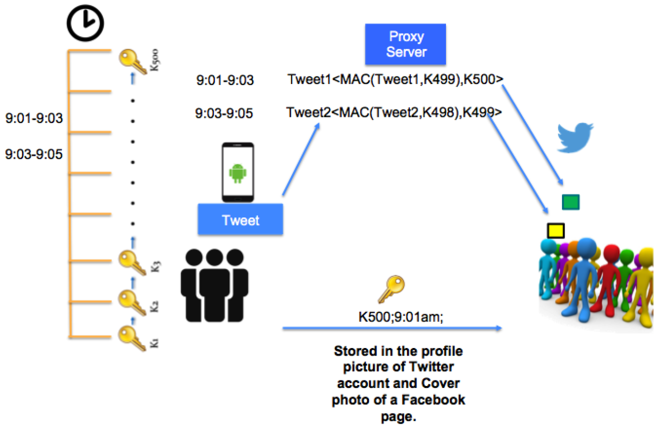
Proxy Server
K_{500}
K_3
K_2
K_1
9:01-9:03
9:03-9:05
9:01-9:03
9:03-9:05
Tweet1<MAC(Tweet1,K499),K500>
Tweet2<MAC(Tweet2,K498),K499>
Tweet
K500;9:01am;
Stored in the profile picture of Twitter account and Cover photo of a Facebook page.

Fig. 3. Time-based cryptographic signatures

B. Bootstrapping Hash Chains on Client

The one way hash chain is represented as pebbles which represent different positions in the hash chain. A hash chain of length n requires $log_2 n$ space and $log_2 n$ computation for generating the next hash key.

SHA2-256 hash function is used for the one way hash chain creation. SHA2 - Standard Hash Function 2 are a set of cryptographic hash functions designed by the NSA (National Security Agency) and are still considered secure unlike its predecessor SHA1 which may have already been cracked. The SHA2-256 version of the family produces 256 bit hashes.
Currently, the hash chain size on the client is $\int_a^b 2^15$ keys. After the hash chain creation is complete, the Super User makes the last key of the chain public. For this purpose, the Super User sends this key and the current Twitter server timestamp to the proxy server. The proxy server in turn posts

this data, encoded in an image, - profile picture of Twitter account and cover picture of Facebook page respectively.

C. Posting content

All content the user intents to post to Social media is routed via the proxy server. The proxy does not store the hash chain on it and hence the client sends the post content along with the message digest to the proxy, which in turn posts it, encoded in a image. The client will not be able to post content to the social media directly because it does not have the OAuth token of the account.

The process is depicted in Figure 2 and 3. and is described as follows:

- The message signature follows a time-based model. The content is signed using the key from the current time interval and along with this Message Authentication Code(MAC), the key from the previous interval is also made public.
- This message along with the metadata, the user identifier token, and the signature generated by signing with the user's private key is send to the proxy.
- The proxy verifies the signature using the public key of the user. [Lookup with the token]. If its valid, then the message is posted to Twitter/Facebook.

D. The Guardian Project

The Guardian Project creates easy to use secure apps, open-source software libraries, and customized mobile devices that can be used around the world by any person looking to protect their communications and personal data from unjust intrusion, interception and monitoring.

The Guardian Project provides an Android library IOCipher for Virtually encrypted disks. IOCipher is based on SQLCipher, an open-source library for encryption of SQLite databases. SQLCipher provides transparent, secure 256-bit AES encryption of SQLite database files. It uses OpenSSL's libcrypto for encryption. IOCipher uses libsqlfs to enable us to use an sql database as a file system. libsqlfs built as a module in FUSE (File System in User Space) to enable applications to access it.

The files stored on the device have to be encrypted with a user provided password to prevent the hash chain information as well as the private key from being misused when the user misplaces or loses his phone.

V. EVALUATION

The android application is tested on Android versions 2.3 to 6.0.

A. Headings, etc

B. Figures and Tables

TABLE I

AN EXAMPLE OF A TABLE

One	Two
Three	Four

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an document, this method is somewhat more stable than directly inserting a picture.

Fig. 4. Inductance of oscillation winding on amorphous magnetic core versus DC bias magnetic field

VI. CONCLUSIONS

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

REFERENCES

[1] M. Jakobsson; RSA Labs., Bedford, MA, USA, "Fractal Hash Sequence Representation and Traversal", IEEE International Symposium on Information Theory, 2002.

[2] Twitter OAuth Reference, https://dev.twitter.com/oauth/overview

[3] Facebook OAuth and Developer Reference, https://developers.facebook.com/docs

[4] Embedded Jetty Server, http://www.eclipse.org/jetty/documentation/current/embedding-jetty.html

[5] Google Developer Guide for Google Cloud Messaging, https://developers.google.com/cloud-messaging/android/start

[6] The Guardian Project - IOCIpher, https://guardianproject.info/code/iocipher/

[7] D. L. Johnson, E. Belding, K. Almeroth, and G. van Stam. Internet Usage and Performance Analysis of a Rural Wireless Network in Macha, Zambia. In ACM Networked Systems for Developing Regions (NSDR) Workshop, San Francisco, CA, June 2010.

[8] D. L. Johnson, E. M. Belding, and G. van Stam, Network Traffic Locality in a Rural African Village, International Conference on