A Mini Project Report

on

Anon-Pass: Practical Anonymous Subscriptions

Submitted by

Aditya Sriram 13IT104
Ashish Singh 13IT107
Rohan Nair 13IT135
Ohileshwar Itagi 13IT251

VI SEM B.Tech(IT)

in partial fulfillment for the award of the degree
of

Bachelor of Technology

In

Information Technology

At

Department of Information Technology
National Institute of Technology Karnataka, Surathkal.
April 2016

Department of Information Technology

National Institute of Technology Karnataka, Surathkal

Information Assurance and Security Project
End Semester Evaluation (April 2016)

Course Code: IT352

Course Title: Information Assurance and Security

Title of the Project: Anon Pass: Practical Anonymous Subscriptions

Details of Project Group

Name of the student	Register No	Signature with Date
1. **Aditya Sriram**	**13IT104**	
2. **Ashish Singh**	**13IT107**	
3. **Rohan Nair**	**13IT135**	
4. **Ohileshwar Itagi**	**13IT251**	

Name of Course Instructor: Mr. Gaurav Prasad

Signature of the Instructor(with date):

Place:

Date:

CERTIFICATE

This is to certify that the project entitled "Anon-Pass: Practical Anonymous Subscriptions" is a bonafide work carried out as part of the course Information Assurance and Security (IT352), under our guidance by **Aditya Sriram, Ashish Singh, Rohan Nair and Ohileshwar Itagi,** students of VIth Sem B.Tech (IT) at the Department of Information Technology, National Institute of Technology Karnataka, Surathkal, during the academic semester 2015-16, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, at NITK Surathkal.

Place: NITK,Surathkal

(Signature of the Instructor)

Date:

DECLARATION

We hereby declare that the project entitled ”Anon-Pass: Practical Anonymous Subscriptions”, submitted by us for the as part of the partial course requirements for the course Information Assurance and Security (IT352), for the award of the degree of Bachelor of Technology in Information Technology at NITK Surathkal during the VIth semester has been carried out by us. We declare that the project has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles elsewhere.
Further, we declare that we will not share, re-submit or publish the code, idea, framework and/or any publication that may arise out of this work for academic or profit purposes without obtaining the prior written consent of the course Faculty Mentor and Course Instructor.

(Signature of the Students)

Place: NITK,Surathkal
Date:

Abstract

Electronic subscriptions are widespread and quickly becoming the dominant mode of access for services like music and video streaming, news, and academic articles. Although electronic subscriptions are convenient for users, they reveal a lot of information, ranging from personal preferences to geographic movements. Many users want electronic services, but they also want privacy. Simply anonymizing data doesn't always protect users' privacy.

Through this project, we attempt to solve the issue of protecting user privacy and simultaneously control admission so that unathenticated users are barred from using the services. Anon-Pass is our solution to combat this issue. It lets users authenticate anonymously while preventing mass sharing of credentials. Service providers can't correlate users' actions but are guaranteed that each account is in use at most once at a given time.

Contents

List of Figures

List of Tables

1 Introduction

With growing demand for online services, there is a great need to protect the information provided by users of such services. In addition to protecting the information, it is also essential to ensure there is no malpractice involved and there is a genuine authentication process involved which ensures that users are able to enjoy all the services by upholding their anonymity and admission control is satisfied too.

It's difficult to create systems that protect user privacy and simultaneously control admission that is, keeping out users who haven't paid. Foregoing one of these two goals makes achieving the other considerably easier. If users are required to log in to an account, foregoing anonymity, a service can enforce that no user is logged in twice simultaneously. On the other hand, a subscription system using a single shared identity for all users prevents user identification. However, subscribing users could share the secret with nonsubscribers. Ideally, we would have an anonymous subscription system that protects the interests of both the service and the users. At a high level, such a service needs two operations: registration and login. Registration lets users signup for the service, at which point they might need to provide identifying details, such as a credit card number or public key. Login lets registered users access protected resources via their subscription. Informally, an anonymous subscription service ensures that users' logins aren't linked to the information they provided at registration and login sessions aren't linkable with one another.

Anon-Pass is intended to instantiate our protocol in a way that's practical for deployment. We present a conceptual framework for the system in which the various system functionalities are separated. There are three major pieces of Anon-Pass functionality: the client user agent, the authentication server, and the resource gateway. The client user agent and the authentication server correspond to the client and server in the cryptographic protocol. The resource gateway enforces admission to the underlying service, denying access to users who aren't properly authenticated. An Anon-Pass session is a sequence of epochs beginning when a user logs in and ending when the user stops re-upping. We depict the most distributed setting, wherein each of the three functions is implemented separately from existing services. However, a deployment might merge functionality,for example, the resource gateway might be folded into an already existing component for session management.

2 Literature Survey

2.1 Background

Ideally, we want each new client operation to appear to be from a new user, unrelated to previous users. However, when two operations are authorized with the same client secret at the same time, it must be clear they're from the same user. To keep login credentials verifiable but also make them changeable, we divide time into equal-length intervals or epochs, agreed on by both clients and servers. Clients use the epoch as input to a pseudorandom function (PRF), which allows them to change the login credential for each epoch. Changing login credentials means each client appears to be a new user in every new epoch, but each client secret can create only one unique login credential per epoch, preventing multiple simultaneous logins with the same credentials.

A login credential provides access only for the duration of an epoch. There's a tension between a service provider's desire for a long epoch (to reduce server load) and users' desire for a short epoch (to improve anonymity). The service needs to perform cryptographic checks during login, making login a computationally expensive operation. Consequently, the service provider wants to maximize epoch length. However, users are unlinked from previous activity only once an epoch boundary has passed and hence prefer a shorter epoch. For example, when listening to a music streaming service, users probably don't want to wait five minutes or even one minute for the next track to play. We believe that users want unlinkability across accesses to distinct pieces of content, such as a movie, song, and news article, but not all accesses to protected resources need to be unlinkable. Hence, we designed Anon-Pass with conditional linkage, wherein some accesses to protected resources are linked to reduce computational cost when privacy is less important. We provide short epochs, giving users the ability to reanonymize quickly if they so choose, while also providing an efficient method for those who don't need unlinkability to cheaply reauthenticate for the next epoch.

An anonymous subscription scheme with conditional linkage consists of two algorithms namely Setup and EndEpoch and three protocols Reg, Login, and Re-up. Setup and EndEpoch are used for bookkeeping of internal service state and let us clearly define service provider operations in our proofs. The three protocols comprise the scheme's primary functionality: registering new users in the system (Reg) and authenticating clients to the system (Login and Re-up). Re-up differs from normal authentication because it requires that the client is already logged in to the system. The protocol implements our scheme's conditional linkage aspect by extending the current user session into the next epoch.

2.2 Outcome of Literature Survey

Ideally, we would have an anonymous subscription system that protects the interests of both the service and the users. At a high level, such a service needs two operations: registration and login. Registration lets users sign up for the service, at which point they might need to provide identifying details, such as a credit card number or public key. Login lets registered users access protected resources via their subscription. Informally, an anonymous subscription service ensures that users' logins aren't linked to the information they provided at registration and login sessions aren't linkable with one another.

Although cryptographic protocols for providing anonymous credentials services already exist, there are problems with putting these protocols into practice. Many of these protocols are designed for thousands of concurrent users, however, Netflix streamed 1 billion hours of content in July 2012 and has millions of subscribers. At this scale, some proposed cryptographic operations require too much computation. Existing work doesn't focus on realistic evaluation scenarios, making it difficult to understand what performance issues would arise in a deployed system. Anon-Pass, a new protocol for anonymous subscription services, aims to achieve significant improvements in efficiency over prior protocols.

2.3 Problem Statement

To build a subscription system model that provides for a legitimate client's authentication and anonymity along with admission control for the system.

3 Methodology

3.0.1 Basic Client-Server Authentication Model

It comprises of two modules namely Registration and Login. This authentication is an example of zero knowledge proof protocol wherein the client authenticates without actually giving the value of the key to the server.

Registration

- A prospective client registers with the server by sending his username and password.
- Upon validation, server and client agree on a secret key ”d”.

Login

- The client computes Yclient = f(d,t) and sends it to the server.
- The server computes Yserver = f(d,t) and stores all the values in a list.
- If Yclient is present in the list, then the user is legitimate and he will be granted access to the web service.
- If not, the user is not genuine and authentication fails.

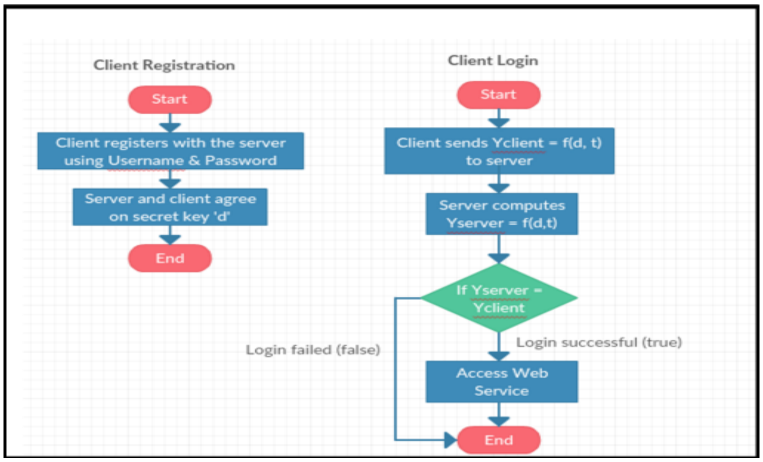
Client Registration
Start
Client registers with the server using Username & Password
Server and client agree on secret key 'd'
End
Client Login
Start
Client sends Yclient = f(d, t) to server
Server computes Yserver = f(d,t)
If Yserver = Yclient
Login failed (false)
Login successful (true)
Access Web Service
End

Figure 1: Basic Client-Server Authentication Model

3.0.2 Kerberos Authentication Model

The steps involved in the working of the Kerberos Authentication System is as follows:

- When the user logs in to his/her machine, the username is sent to KDC server for login and the KDC server will provide the Ticket Granting Ticket (TGT) in return.
- KDC server searches the username in the database, on finding it, a TGT is generated by the KDC which will be encrypted by the users public key and sent back to the user.
- When the user gets the TGT, the user decrypts the TGT with the hep of his private key.
- The TGT received by the client from the KDC server will be stored in the cache(Kerberos tray) for use for the session duration. There will always be an expiration time set on the TGT offered by the KDC server, so that an expired TGT can never be used by an attacker.
- Now the client has got TGT in hand. If suppose the client needs to communicate with some service on that network, the client will ask the KDC server for a ticket for that specific service with the help of TGT.

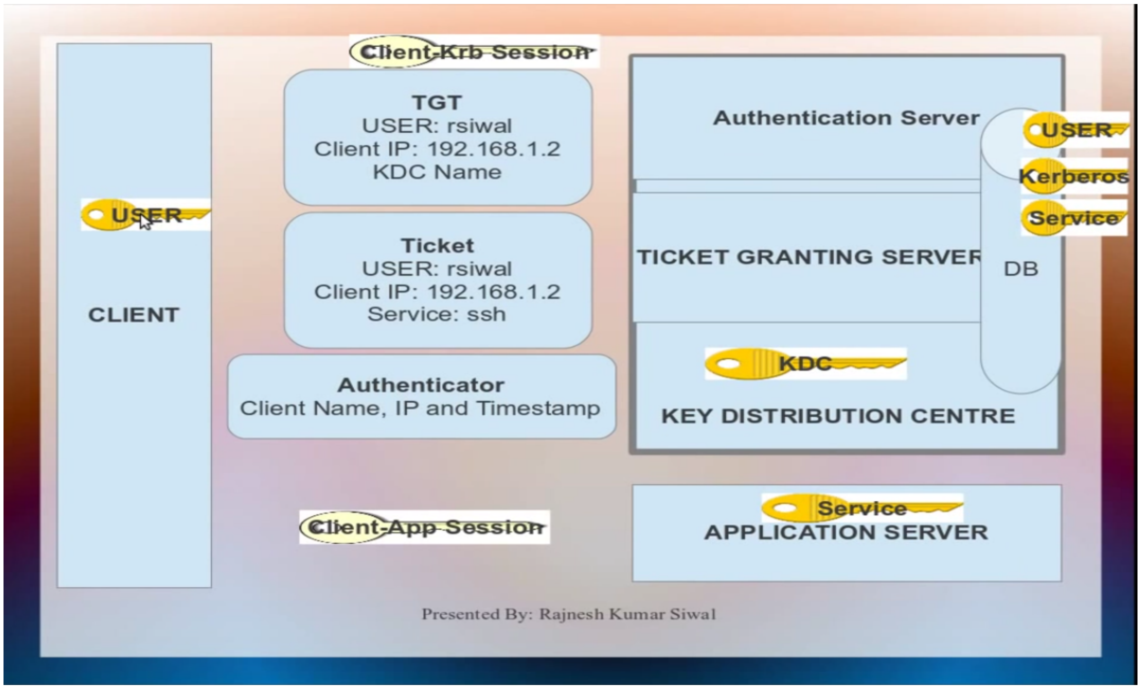

Figure 2: Kerberos Authentication Model

3.0.3 Anon-Pass Authentication Model

The communication between the authentication server, resource gateway, and client user agent with respect to the client and the service is realised as shown in Figure 3. The steps followed in ensuring a system model that upholds anonymity and admission control are as follows:-

- The user agent initiates communication.
- The authentication server verifies the credentials and returns a sign-in token to the user agent.
- The user agent communicates this sign-in token to the resource gateway.
- It then passes this information to the client application.
- The client application includes the token as a cookie along with its normal request.
- The gateway checks that the sign-in token hasn't already been used in the current epoch and then proxies the connection to the application server.
- The application server returns the requested content.
- The gateway verifies that the connection is still valid before returning the response to the client.

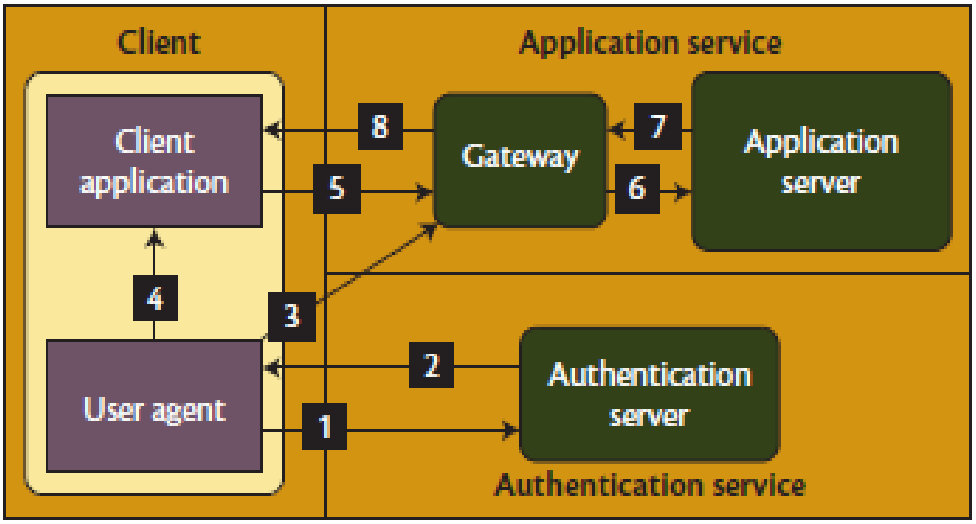

Figure 3: Anon-Pass Authentication Model

4 Implementation

4.1 Work Done

We implemented the Anon-Pass authentication model using the PyWebSocket module provided in Python libraries. The client side is implemented as a web browser and the server is a python script. The client (web browser) is realized using HTML, CSS and JavaScript. The communication between the client and the server is facilitated by web sockets. Messages at the client side are printed on the console and those at the server side are printed on the terminal.

There are three components to this authentication model. Namely, registration, login and re-up.

For registration, the client sends the username and password to the server. In response, the server sends a key 'd'. The server will also store this 'd' value.

For login, the server sends the current epoch number (t) to the client. The client computes the value of Yclient using the function f(d,t). The Yclient value is then sent to the server. The server, refers to its list of keys and computes the Yserver values for each of them using the function f(d, t). If the Yclient value is present in this list, the the client is a legitimate one. The server updates the clients connection status and sends it a token which client will use to access the web service.

For re-up, the clients which have already logged in send their Yclient values to the server and the server retain their Yclients values after checking their connection states and deletes the rest of the connection states.

4.2 Results and Analysis:

We simulated three different scenarios: a baseline system with basic client-server authentication, a kerberos authentication system in which users couldn't reauthenticate cheaply, and the full Anon-Pass system including re-up. When authentication was involved, we used an epoch length of 10 seconds.

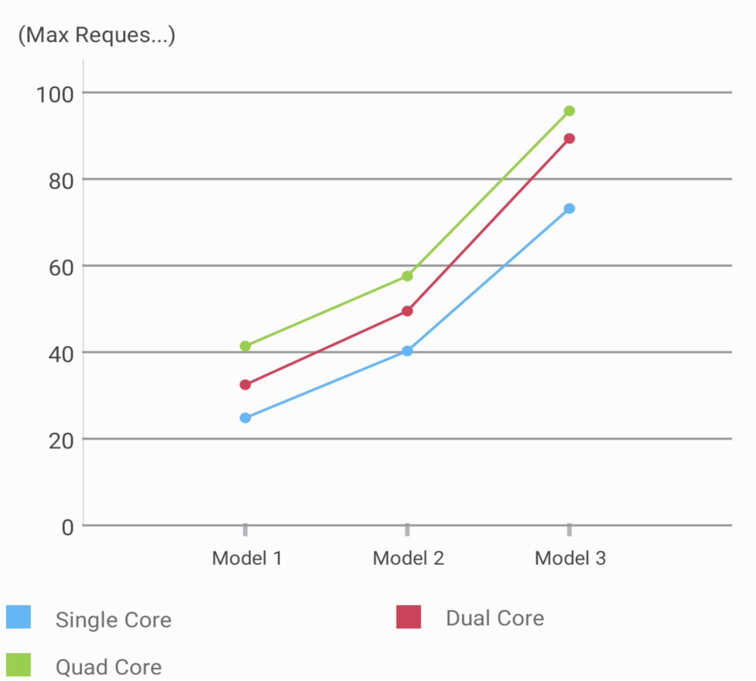

Figure 4: Performance Measure

	Model 1	Model 2	Model 3
Single Core	25	40	73
Dual Core	37	53	89
Quad Core	41	59	96

Table 1: Performance Measure on the basis of cores

Model 1 represents the basic client server authentication system.
Model 2 represents the Kerberos Authentication System.
Model 3 represents the Anon-Pass Authentication System.
From the above graph, we can infer that the Anon-Pass authentication system perform significantly better than the other two systems. Hence, it is much more scalable.

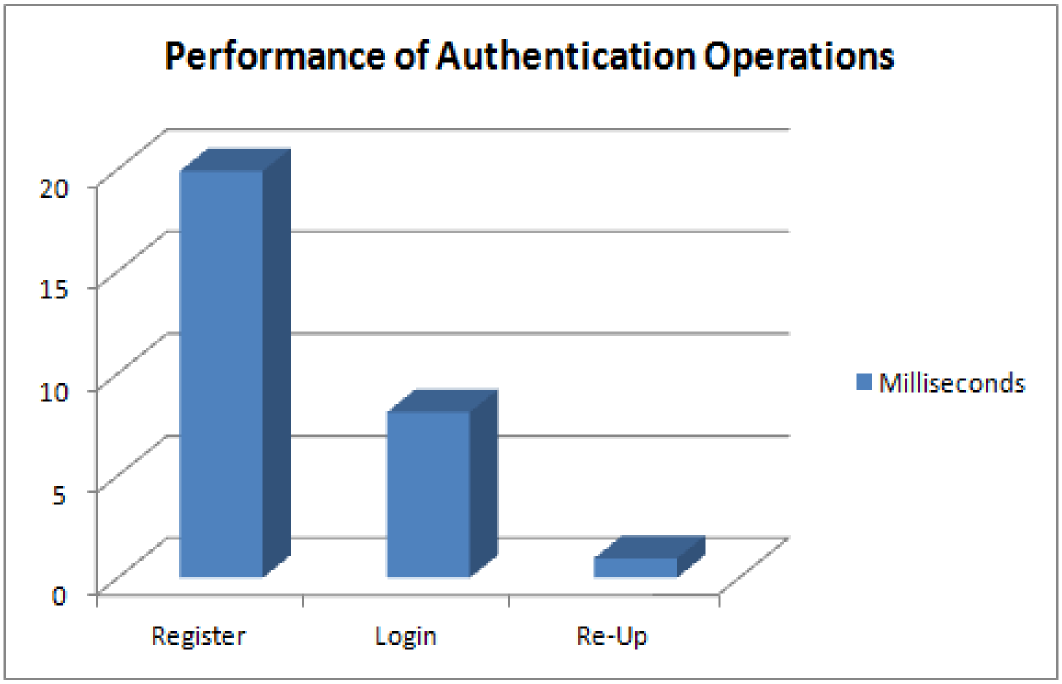

Figure 5: Performance of Authentication Operations

We can infer from the above graph that the Anon-Pass system with Re-Up functionality performs better than a system with purely register or purely login functionalities. This is because Re-Up protocol is computationally less expensive than Login protocol.

5 Conclusion

Through this project, we aim to compare and contrast three different authentication methods namely:

- Basic Client-Server Authentication
- Kerberos Authentication
- Anon-Pass Authentication

After an extensive literature survey, we arrived at a conclusion that the basic authentication models suffers from a wide range of drawbacks which include high cryptographic computational cost, low flexibilty and the possibility of multiple logins. The Kerberos model suffers from a few drawbacks as well which is its increased dependency on the Key Distribution Centre (KDC) for all activities pertaining to its operation and the issue of pseudo-anonymity.

To address the issues of the Basic Authentication Model and the Kerberos Authentication Model, we developed the Anon-Pass Authentication Model which takes care of admission control and anonymity.

6 Future Work

The Anon-Pass System has been developed as a Proof of Concept Application. We intend to scale the application to realise it in a full fledged commercial subscription system.

References

[1] Michael Z. Lee, Alan M. Dunn,Jonathan Katz, Brent Waters, Emmett Witchel,”Anon-Pass: Practical Anonymous Subscriptions”,IEEE S&P Symposium(2014), 20-27.

[2] Jan Camenisch, Susan Hohenberger, Markulf Kohlweiss, Anna Lysyanskaya,Mira Meyerovich,” How to win the clone wars: Efficient periodic n-times anonymous authentication”, ACM Conference on Computer and Communications Security(2006), 201–210.

[3] B. Clifford Neuman,Theodore Ts‘o,”Kerberos: An Authentication Service for Computer Networks”,IEEE Communications Magazine(1994),33-38.