Secure End User E-Mail Client

Akash singh
akash4393@gmail.com

Abhijeet Ranadive
abhi.ranadive@gmail.com

September 13, 2015

Abstract

From list of groceries to next acquisition of shares, most modern day communications relies on the service provided by electronic mail. It's almost given that service providers take extensive care of communication layer security and safely storing user data. But, who protects the privacy of a user from the service provider itself? What if they are compromised? What if they decide to start going through your mail? Do we still consider that secure? Do we still have our privacy? Well, as absurd and far fetched as it sounds, most service providers have been caught or have openly admitted to doing so.

With this project we aim to provide a true sense of privacy where, nobody apart from the intended party can read and access the messages. This application will use modern encryption tools to patch a massive security hole in the most widely used communication platform.

1 Problem Statement & Motivation

"When you upload, submit, store, send or receive content to or through our services, you give Google (and those we work with) a worldwide license to use, host, store,reproduce, modify, create derivative works (such as those resulting from translations, adaptations or other changes we make so that your content works better with our services), communicate, publish, publicly perform, publicly display and distribute such content." *-Google*[1]

Widespread usage of e-mail services with such absurd privacy policies has been our biggest motivation for developing a solution to protect the privacy of users. If the most trusted and used email provider is guilty of partaking in such privacy breaches, how can a user be comfortable sending sensitive/private data without a better solution. Even though till now there has been no such reported incidents where a provider like Google was guilty of any malicious practice, we can never be sure that this wouldn't change. And it's not only a privacy breach when such a service provider is hacked or decides to sell user information, but also when it participates in programs like PRISM, which is used for mass surveillance of users by the government itself.

2 Solution

Our solution is to add another layer of security in the existing model by encrypting the body of an email. By doing so, we not only continue to reap benefits of security measures taken by email service providers, but also protect the user conversations from the provider itself. If any security breach occurs at the service providers end, the confidentiality of data will never be compromised. Since our solution will be utilizing RSA key pairs for successful encryption and decryption of conversations, the only way an adversary can access the text of an email is by breaking RSA itself (which is not considered an imminent threat). Along with adding a layer of protection, the solutions also works as a digital signature, ensuring the authenticity of an email.

2.1 Standalone Client

We will develop a fully functional email client in Java, that will utilize the PGPHelper class, along with RSA keys for encrypting and decrypting emails. This client will also implement HMAC based data integrity and authentication scheme. The client will maintain a

database for storing contacts along with their public keys, which will be used for establishing a secure communication. The client will require initial setup/handshaking for acquiring the public keys of both parties.

2.2 Browser Extension

A browser extension will use a different approach for providing the same solution where the email conversation will be secured using PGP (Pretty Good Privacy) standards along with HMAC (Hash-based Message Authentication Code) for maintaining data integrity and authenticity.

The two different approaches described here will be put to test in order to find out as to which provides a better overall security to the users.

3 Implementation Details

3.1 Platform

Java Virtual Machine (JVM)
Cross platform since JVM is supported on Linux, Unix, Windows and Android Operating Systems

3.2 Language

- *Java Programming Language*
- *Structured Query Language*

3.3 Components

- *Java Cryptographic Libraries*
- *RSA key pairs*
- *mySQL Database*

4 Timeline And Work Distribution

4.1 Timeline

- *1st October, 2015*
    1. *User Login authentication and send email set-up*
    2. *UI for Send Email with appropriate boxes for To, Cc, Bcc, Subject and Body field.*
- *1st November, 2015*
    1. *Receive Mechanism Set-up*
    2. *UI for receiving email*
    3. *Database connection(will attempt to sync google contacts)*
    4. *Key pair generation, encryption and decryption*
- *1st December, 2015*
    1. *Completion of email client with UI*
    2. *Secure database with public key entries*
    3. *Testing and bug fixing*
- *10th December, 2015*
    1. *Final rounds of testing*
    2. *Final Report Completion*

4.2 Work Distribution

Work distribution has not yet been finalized. We will be following an Agile Methodology for the development phase (namely Extreme Programming) and individual contribution can be tracked in real time as we will be setting up a public story board (on trello.com) and adding a link to it on our Github repository.

References

1 Google End User License Agreement. *http://www.google.com/intl/en/policies/terms/*

Wikipedia PRISM, NSA. *https://en.wikipedia.org/wiki/PRISM_(surveillance_program)*

The Guardian Google: don't expect privacy when sending to Gmail *http://www.theguardian.com/technology/2013/aug/14/google-gmail-users-privacy-email-lawsuit*