Beeswax
a platform for private web apps

Jean-Sébastien Légaré*, Robert Sumi and William Aiello
UBC NSS Lab

The University of British Columbia

Presented at PETS'2016, July 20th 2016
Frederic Nietzsche  
today at 11:04 AM

Messages you send to this chat and calls are now secured with end-to-end encryption. Tap for more info.

We should consider every day lost on which we have not danced at least once. And we should call every truth false which was not accompanied by at least one laugh.

11:03 AM

Chat with kamicut
Chat  Edit  View  Help

kamicut is online.

kamicut (ludwig)  11 May, 9:47 pm
OK, here’s the finished report!

kamicut (ludwig)  11 May, 9:47 pm

nadim (Windows Laptop)  11 May, 9:48 pm
Awesome, thanks!

nadim (Windows Laptop)  11 May, 9:48 pm
Let’s grab sushi later.
Are they secure? Is it really Private?

Google Hangouts/GTalk glitch sends chats to wrong recipients

[UPDATE] Be careful what you say or be sent to the wrong person. Google is looking into it.

By Michael Lee | September 26, 2013 -- 07:28 GMT (00:28 PDT) | Top

Sharing Links On Facebook Not As Private As You Might Think

BY DAVID MURPHY | JUNE 11, 2016 02:50PM EST | 0 COMMENTS

It’s not very hard for developers to extract links from Facebook’s database, and these links might have sensitive information right in the URL.
Ex: Facebook Messaging, cryptocat, google talk otr.

How could one gain assurance?

- Audit the code?
- Rely on conclusions of a diligent self-identified community of experts?
- Do it again for every app?
Root of the Problem

• Client-Side code of private apps contains private information – The keys.
• W/O containment of keys, plaintext, crypto functionality, app must be in TCB
• Moreover, *every* app performing end-to-end security must be trusted

+ TCB of all applications
Beeswax

- A security platform to reduce the TCB of private web applications.
- Disaggregation, and containment of security-critical data & functionality.
- Sharing of this functionality provided in well-defined APIs.
- Allows scrutiny to be focused on the platform (Instead of every app)
- Implemented as a Google Chrome (v40) extension, (5K lines of code)
- Deployable now and allows rich web application development
TCB Grows Larger

Vulnerabilities in application code can exfiltrate data.

Must be in TCB

Repeats for each app.

Vulnerabilities in application code can exfiltrate data.
UI
App. Logic
Ciphertext Msgs
Plaintext
Layout / Style

UI
App. Logic
Ciphertext Msgs
Plaintext
Layout / Style

Key Mgmt. / crypto
Runtime / APIs

Protect Keys
- Move them to platform
- Application gets key handles

Must be in TCB
Needs no trust
UI
App. Logic
Ciphertext Msgs
Key Mgmt. / Crypto
Runtime / APIs

Must be in TCB
Needs no trust

UI
App. Logic
Ciphertext Msgs
Layout / Style

Plaintext Viewer

Also protect plaintext

- Provide opaque handles to the application
- Challenges:
  - Keep look n feel
  - Maintain current dev practices.
Challenge - Isolating plain text

Plaintext isolated in “private areas” taken in charge by Beeswax.

1. API call designates region of DOM to display confidential info.

2. Platform protects region of DOM from access by page JS.

3. API call to display and inputs ciphertext in/out private area.

**Isolation uses ShadowDOM [W3C], similar to ShadowCrypt [CCS2014].

**We perform JS environment changes to protect access and allow events.
Beeswax isolates keys and plaintext
Isolated data cannot be exfiltrated

Are we done?
Can we turn the app stacks blue?

Must be in TCB
Needs no trust
UI
- App. Logic
- Ciphertext Msgs

Runtime / APIs
- Ciphertext
- Layout / Style

Plaintext Viewer
- Plaintext

Key Mgmt. / Crypto
- Plaintext

Must be in TCB
- Needs no trust

NO!
Blue means we must assume app can be malicious.
A malicious app can spoof the UI.
Challenge – Defeating UI spoofing by app

Application may or may not use Beeswax APIs.

App might try to provide its own “privacy” markers

E.g. “Bob’s in the ‘To:’ field. Is this message really being sent to Bob?”

Application could show “green locks” or “green borders”, but can’t be trusted.

Beeswax could change the page to add indicators, but the app controls the window.
Beeswax Privacy Indicator

We add an indicator of privacy in an *unspoofable* region of the tab.

User interactions in private areas toggle the privacy indicator.

Tells if DOM region of interest is private

Content is hidden from the app

Events locked* to region

User kb + mouse Interactions
- Keys/plaintext are unavailable to the application
- P.I. indicates where events and text go.
- In TCB: Beeswax YES, Apps: NO

- Must be in TCB
- Needs no trust
Split Functionality: Platform and App

Application provides functionality and takes care of sharing. Intention: “User wants to write a message to Bob+Carl”

Platform manages keys and identities.

Platform establishes secure end-to-end data streams between users. (crypto).

When interacting with a private area, the platform allows the user to verify true recipients of a message.
Transparent Key Management and Distribution

Beeswax has Built-in key management:

- Automatic distribution of Public Keys
- Key Agreement Protocol between pairs of users ("friendships")
- Symmetric key crypto API ("streams")
Beeswax Identities

At setup, a user’s Beeswax browser extension will generate 2 keypairs (sign, encrypt) and post a self signed cert of both to a configured twitter account.

The Beeswax background process in the extension monitors and reposts certs periodically.

*A similar process allows users to retrieve and monitor friend’s keys based on twitter IDs
Beeswax Key Distribution

The application initiates friendships with other users. (Triggers the KAP).

Beeswax periodically monitors online certs against those in DB. Handles revocation.*

Fetch friend @bob’s

@bob’s certs

Beeswax @alice (background)
Key Agreement -- Friendship Channel

Key-Agreement-Protocol (KAP) creates secure bi-directional control channel between pairs of users, Friendship Channel. E.g. used for invitations and exchanging key information (see below).

**API get_friend(@accountid) -> friendship**

Establishes a set of symmetric keys used for secure communication of app signalling, such as invitations to streams.
Streams

Streams are media channels. Stream creators can invite other users over friendship channels.

**API invite(<friendship>, <streamid>) -> invitation**

Invite participants to a stream by messages over friendship channel

**API accept_invite(<invitation>) -> streamid**

Application receives a key handle for this stream (handle to a symmetric key).

Application relays ciphertext attached to streams.
Evaluation

- Mechanisms fit for the development of modern web application
- Transformed existing web communication application (IRC) to support encrypted messaging between groups of users
- Created new encrypted photo gallery to demonstrate ability to handle richer media types
- Acceptable performance
Evaluation - Encrypted IRC Client

Adding encrypted messages to an IRC client:

Beeswax users can create encrypted IRC channels

Modified KiwiIRC v0.9.0: 400 LOC added to client-side (7%)
Evaluation - Secure photo sharing (PicSure)

Regular tools: jquery, bootstrap, node.

Richer media type support: private areas supporting images (Beeswax photo chooser)
Performance - Microbenchmarks

Takeaways:

• Encryption cost is predictable, linear with plaintext size.

Re: runtime

• ~52 ms average page load increase

• 2.5x slower event processing to sanitize events from confidential information in private areas
Why just the web. What about mobile?

- Android OS does not have an architecture for secure modules to be loaded
- No allocation for an unspoofable area of the screen (privacy indicator)
Discussion

• Provides protection against exfiltration by the application provider

• Like any platform, features can be added as platform matures

• Key distribution easy and automatic, deployable now.

• Focus scrutiny on platform, not apps

Platform and apps are open source, available on github:

https://web-priv.github.io/beeswax/
END OF RIBBON.
RESERVE SLIDES FOLLOW
Privacy indicator states

Fig. 2. The Privacy Indicator “unprotected” (left), “protected” due to keyboard events (“K”) in a private area (middle), and showing a security warning (right).
Other spoofing

Talk about other ways to spoof there?

- Lying about recipients
- Overlaying elements
- Stealing events from private areas
- Locking mechanism

Refer to paper?
Beeswax Identity Management

Users are registered to a Pub/Sub service

Users verify binding between P/S account ID and person they want to communicate with

Only account owner can post to that account
famous cats

astronaut cat