Proforma

Original aims

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio, Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, and the whole LaTeX-community for support, ideas and some great software.

Work completed

Nicolas Repp, Hinrich Harms, and the whole LaTeX-community for support, ideas and some great software.

Special difficulties

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, and the whole LaTeX-community for support, ideas and some great software.

Declaration

I, Christopher Louis Harding of Queens' College, being a candidate for Part II of the Computer Science Tripos, hereby declare that this dissertation and the work described in it are my own work, unaided except as may be specified below, and that the dissertation does not contain material that has already been used to any substantial extent for a comparable purpose.

Computer Science Tripos Part II Project, March 2011

C. L. Harding

Contents

List of Figures

List of Tables

Acronyms

DRY	Don’t Repeat Yourself
API	Application Programming Interface

UML	Unified Modeling Language

Acknowledgments

Put your acknowledgments here.

Many thanks to everybody who already sent me a postcard!

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio, Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, and the whole LaTeX-community for support, ideas and some great software.

1 Introduction

Facebook is most common social network with X million users and Y daily hits. Currently all communication in plaintext. Single sentence description of project and what it does. Single sentence definition of broadcast encryption.

1.1 Motivation

- General context - recent media coverage of Facebook, privacy and security concerns. Data (user passwords) stolen from web sites by hackers. Privacy a big issue for the 21st century?
- Context in the field of Computer Science - new social networking services being launched (Diaspora etc.). Networking effects and Facebook user base make it hard to compete. Idea : keep using Facebook, add encryption on top.
- Limitations - [1] does not hide the social graph. Arguably this is Facebook's real asset.

1.2 Existing work

- Symetric only schemes:
  - FireGPG - `http://blog.fortinet.com/encrypting-facebook/`
  - CryptFire
  - TextCrypt - `http://subrosasoft.com/OSXSoftware/index.php?main_page=product_info&cPath=210&products_id=207`

Lots of these around.

- Complete schemes:
  - uProtect.it
  - flyByNight - `http://hatswitch.org/~nikita/papers/flybynight.pdf`

2 Preparation

2.1 Approaches

- Approaches in existing work
- Model of where the app could sit
  - Proxy server (remote)
  - Facebook application (client JavaScript)
  - Bookmark button (client JavaScript + remote state on server)
  - Greasemonkey (client JavaScript)
  - Extension (client JavaScript + native code + local state)
- Why good/bad? Try to be quantatative
- Leads on to lanugage and library choices
- How to call C++ from JavaScript (binding, linking, compilation, marshalling).
- Facebook Graph API (JavaScript API etc.)

2.2 Broadcast encryption

- Naive implementation
- More advanced schemes (and why I don't use them)
- Key management and size (NIST recommendations)

2.3 Storing data in images

- Description of Facebook/JPEG compression process (problem statement)
- Analysis of theoretical capacity
- Evaluation of naive implementations (MATLAB demonstrations)
  - Completely naive (encode in RGB values)
  - Even less so (encode in DST with bit masks)
- Possible approaches (refer to research on Haar stuff etc.)

2.4 Testing plan

- What kinds of testing will I use?
- How can I make these tests possible?
- Test bed or framework that needs to be in place

2.5 Proffesional practice stuff

- Software development model
- Version/source control
- Performance bounds

2.6 Requirements analysis

- Functional and non-functional
- Same as success criteria, but *derive* them
- Possible requirements:
  - it works (for small groups)
    - must be able to do X,Y,Z.

    * encryption must be sound (refer to NIST etc.)

- it *would* work for lots of people
    * image method must be modular - allow choice max capacity, future upgrade.
    * UI and text input must also scale to X number of recipients
- the above without affecting normal FB/browsing
    * doesn't add massive timing loads
    * doesn't introduce any security holes
- the above and doesn't negatively affect people without the extension.

3 Implementation

3.1 Project overview

- UML diagrams
- Class diagrams
- Orchestration diagrams
- Directory structure
- JavaScript extension structure

3.2 Interfacing with Facebook

- Using the Graph API
- Obtaining access tokens
- Generating and submitting forms
- Through iFrames

3.3 Modifying the Facebook UI

- Inserting submission controls
- Retrieving content automatically

3.4 Encoding decoding data

- Encryption and decryption
- *Keeping key material safe*
- Forward error correction
- UTF8 encoding/decoding
- Text steganography

3.5 Storing data in images

- Abstractions
- Using the Haar wavelet transform
- Using upsampling
- Using bitmasks on DCT coefficients

3.6 Testing

- Unit
- Regression
- Black box
- White box
- Integration
- Security/penetration testing?

4 Evaluation

It works

Cognitive walkthrough Creating a group of users Migrating profile information Public key management Text submission and retrieval Image submission Image retrieval

! MAYBE - just do one walkthrough with lots of users ! *! OR - split walkthrough in to two halves !*

It would work if deployed large scale

Hypothesis - Upsampled3 is the best method Testing - MATLAB or actual code (or both??) How long it takes Capacity Error rate Conclude - yes Upsampled3 is the best method

Another cognitive walkthrough? Or extend the previous one? - obviously with much more recipients 500

The above, plus doesn't affect normal FB use or browsing

Hypothesis - something about the time Testing - How long does each bit take Not including the image bit, already done Break down of what takes how long Conclude - look at literature on perceptual waiting loading times, its all fine.

The above, plus doesn't affect other FB users without extension

Hmmmmmmmmm?

5 Conclusion

Success criteria a) it works (for small groups) i) must be able to do X,Y,Z. - yes look at cognitive w/through. ii) encryption must be sound - theoretically yes. no guarantee I'm not leaking key information - show where I've tried b) it *would* work for lots of people i) image method must be modular - allow choice max capacity, future upgrade. - yes look at how I compared them. ii) UI and text input must also scale to X number of recipients - yes look at cognitive w/through. c) the above without affecting normal FB/browsing i) doesn't add massive timing loads - yes look at timing breakdown. ii) doesn't introduce any security holes - ************NOT SURE************ d) the above and doesn't negatively affect people without the extension. - ************NOT SURE************

Retrospective Future work Potential deployment (banter)

Bibliography

[1] Özge Aksın et al. **?**Effect of immobilization on catalytic characteristics of saturated Pd-N-heterocyclic carbenes in Mizoroki-Heck reactions**?** In: *J. Organomet. Chem.* 691.13 (2006), pp. 3027–3036.

Appendix

A Graph API

Lorem ipsum at nusquam appellantur his, ut eos erant homero concludaturque. Albucius appellantur deterruisset id eam, vivendum partiendo dissentiet ei ius. Vis melius facilisis ea, sea id convenire referrentur, takimata adolescens ex duo. Ei harum argumentum per. Eam vidit exerci appetere ad, ut vel zzril intellegam interpretaris.

Errem omnium ea per, congue populo ornatus cu, ex qui dicant nemore melius. No pri diam iriure euismod. Graecis eleifend appellantur quo id. Id corpora inimicus nam, facer nonummy ne pro, kasd repudiandae ei mei. Mea menandri mediocrem dissentiet cu, ex nominati imperdiet nec, sea odio duis vocent ei. Tempor everti appareat cu ius, ridens audiam an qui, aliquid admodum conceptam ne qui. Vis ea melius nostrum, mel alienum euripidis eu.

A.1 Appendix Section Test

Ei choro aeterno antiopam mea, labitur bonorum pri no. His no decore nemore graecis. In eos meis nominavi, liber soluta vim cu. Sea commune suavitate interpretaris eu, vix eu libris efficiantur.

Nulla fastidii ea ius, exerci suscipit instructior te nam, in ullum postulant quo. Congue quaestio philosophia his at, sea odio autem vulputate ex. Cu usu mucius iisque voluptua. Sit maiorum propriae at, ea cum primis intellegat. Hinc cotidieque reprehendunt eu nec. Autem timeam deleniti usu id, in nec nibh altera.

A.2 Another Appendix Section Test

Equidem detraxit cu nam, vix eu delenit periculis. Eos ut vero constituto, no vidit propriae complectitur sea. Diceret nonummy in has, no qui eligendi recteque consetetur. Mel eu dictas suscipiantur, et sed placerat oporteat. At ipsum electram mei, ad aeque atomorum mea.

Ei solet nemore consectetuer nam. Ad eam porro impetus, te choro omnes evertitur mel. Molestie conclusionemque vel at, no qui omittam expetenda efficiendi. Eu quo nobis offendit, verterem scriptorem ne vix.

B Project Proposal

Introduction and Description of the Work

Facebook is a social networking service that, as of July 2010, has over 500 million users worldwide. Many people have recently become increasingly worried about Facebook's rather relaxed attitude towards the privacy of personal data. However, attempts at building more secure social networks with technical solutions that ensure data privacy, such as encryption, have not enjoyed much success because Facebook capitalises on the network effect of everyone else using it.

It would be very useful if we could still use Facebook, but encrypt all the data stored there, enabling only those who use the same tool (and possess the appropriate key) to see the plaintext. Obvious targets for encryption are profile information, pictures and comments or other public messages exchanged between users. Extensions might include encrypting videos, events and other information. Ideally, the system would be as unobtrusive as possible; encryption/decryption options should be integrated as if they were implemented by Facebook itself.

Several possible approaches exist for this project. One approach would be to develop a standalone extension for the Firefox web browser (or possibly some other extensible browser such as Chrome). A second would be to develop a 'userscript' for Greasemonkey, an existing Firefox extension. This approach would afford some cross browser compatibility since userscripts are gaining limited support on browsers other than Firefox. Both these approaches would be a form of augmented browsing, relying on modifying web pages on the fly just before they are displayed.

A further (and potentially much more complicated) method would be to develop a complete client. This could be either a web-based or desktop client. In either case, creating a fully functional

Facebook site clone would likely be beyond the scope of this project - however developing a cut down version should at least be considered. The project's first goal would be to assess the suitability of each of these approaches.

Starting Point

Existing experience writing web pages with HTML and CSS. Awareness of JavaScript and tools such as Greasemonkey. Some aspects from the Security courses in Part IB and Part II Computer Science will likely be required.

Substance and Structure of the Project

The project can be broken down into the following main sections. We assume here that the implementation takes the form of the plugin, though as mentioned previously this may not ultimately be the case.

1. Research each of the possible methods of implementation. Choose the most suitable approach.
2. Implement a method of submitting encrypted data to Facebook. Any encrypted data needs to be recognisable as such, e.g. via some kind of tag. Targets for encryption would be photographs, comments and profile information.
3. Implement a method of recovering and displaying encrypted data.
4. Implement a method of key exchange and storage between extension users.
5. Modify the Facebook user interface so that recovery and display of encrypted data happens seamlessly. Ensure an appropriate response for encrypted data for which the user does not have a key.
6. Modify the Facebook user interface so that encrypted submission and key exchange can be done seamlessly by the plugin user.

7. Extend the plugin to improve interaction with users who do not have the plugin installed. Allow the creation of appropriate default behaviors for communicating with users who do/do not themselves have the plugin (which conversations should be encrypted and which shouldn't). Recognizing certain actions and prompting the user may be necessary. Another example - making tags marking content as encrypted more human readable, rather than just perceived gibberish.

8. Demonstrate the plugin by creating a sample set of profiles and performing a set of test actions successfully. Document and record the results.

9. Record various loading times and analyse the performance of the extension.

10. Perform a cursory analysis of the plugins theoretical running time on various actions, with regard to input length and number of users. Demonstrate (as much as possible) that the plugin would be viable for large scale adoption, taking into account the number of Facebook users worldwide.

11. As a possible extension, implement more extensive user interface alterations to change the aesthetic of the encrypted Facebook user experience (e.g. different colour schemes, more tightly integrated, inline icons/controls). This would increase ease of use and make it more immediately clear to any user whether or not they have the plugin enabled.

12. As a possible extension, implement and/or demonstrate compatibility across a range of platforms. Several browsers (other than Firefox) have limited support for userscripts, for example. If writing a standalone plugin, this could perhaps be ported to other browsers (e.g. Chrome, Opera) or operating systems (e.g. Android, iOS).

13. As a possible extension, create a complementary Facebook Application that allows combining encryption options with Facebook's existing privacy controls (among other possible improvements).

14. As a possible extension, extend encryption beyond just comments, photos and profile information. Possibly interesting features might be completely encrypted profile creation (including full name); encrypted events and attendees; encrypted 'pages' and 'likes'; encrypted dates and locations.

15. As a possible extension, look at incorporating steganography techniques (hiding encrypted data in pictures or videos, for example). This might not only clean up the user experience for non-extension users, but preempt any preventative measures Facebook might take to block use of the plugin.

16. Repeat any analysis (particularly of performance) for any completed extensions, as required.

Resources Required

None.

Success Criterion

For the project's core functionalities, each of the following requirements should be met. For any completed extensions, discussion should at least be made on whether the requirements are met, can/could be met with further development, or otherwise.

1. The plugin should be able to perform the set of initial test actions on a set of purposely created test profiles, demonstrable by annotated screenshots. The test actions should provide evidence of successful submission and recovery of photos, comments and profile information, as well as key exchange.

2. The encryption scheme used should ensure at least confidentiality of data and should be immune to any brute force decryption attack.

3. Assuming the previous requirement is met; under analysis, the plugin should perform within acceptable limits for the majority (greater than 95%) of target users in regard to page loading times. A reasonable definition of acceptable limits should be used (e.g.

http://www.useit.com/papers/responsetime.html). Target users are defined as those capable of installing the plugin, thus accurate statistics on typical connection speeds for Facebook users (not including those on mobile devices who would not be able to use the plugin in any case) should be investigated.

4. Analysis of the plugin's operation should demonstrate, superficially at least, that the schemes used would scale up if adoption took place among groups of users larger than the small number of test profiles. If required, define large scale adoption as use among at least 1% of worldwide Facebook users. This will likely require some research into Facebook limitations on, for example, the length of text inputs.

Timetable and Milestones

October 25th - November 1st

Complete a skeleton project with all required sections. Set up version control and review any other library/programming requirements that need to be considered before coding can begin.

If required, begin the process of setting up a certification authority service. Make an initial indication of what schemes will be used for encryption/decryption, key exchange and authentication

Create a prototype Greasemonkey userscript that interacts somewhat with Facebook. Test Greasemonkey's limits, particularly on storing data persistently when browsing from page to page and fetching/parsing additional pages. Repeat this process with a simple Firefox (or alternative browser) extension. At this point it should become clear which approach (userscript, extension or full client) will be most suitable.

Milestones: Project skeleton complete. Two working test applications (Greasemonkey and standalone plugin) that demonstrate simple interaction with Facebook.

November 1st - November 15th

Many possible extensions have been stated for this project - here initial research into their feasibility should be performed.

By the end of this period, a prototype implementation which can encrypt and decrypt text fields (i.e. comments and profile information) will be complete. At this point the user will need to manually select fields for encryption/decryption and supply the appropriate key.

Milestone: First working prototype in place.

November 15th - December 3rd (end of Michaelmas term)

Encryption should be extended to images as well as text fields.

Some automation added to the recovery process. The system should now parse the page and work out what elements may be decrypted. The user will still have to supply the appropriate keys manually.

Milestone: Second working prototype completed, as described.

December 4th (Winter vacation starts) - December 25th

The prototype should now be extended to manage keys automatically. If a CA service exists/is needed then the software should interface with it appropriately. Key exchange hasn't yet been integrated into the browser however.

Recovery of elements can now be done in complete autonomy - we can parse what needs to be recovered, work out what can be recovered, then do so. Again, at this stage, no changes have been made to the Facebook web interface.

Work should begin on the first two written sections of the Dissertation (Introduction and Preparation).

Milestone: Third prototype with working authentication and secure key exchange.

December 26th - January 17th (Winter vacation ends)

During this period modifications should now be made to the Facebook UI to integrate actions into the web page itself. Modification do not need to be attractive (that is left for a later possible extension) but all possible actions should now be able to be initiated through the Facebook site

By the end of the vacation there should exist a draft of the Introduction section and the contents of the Preparation section should be mapped out.

Milestone: Fourth prototype with all core functionality complete.

January 18th (Lent term begins) - January 25th

Polishing of the final application should be made. Informal testing and any necessary tweaks/optimizations should be completed. Useability improvements implemented, e.g. settings and configurations options should be added for default behaviors. Tags should be redone in a more human readable form.

Introduction and Preparation sections should be complete and work should be underway on the Implementation section.

January 26th - February 18th

During this period, any extensions should be implemented. The Implementation chapter should be nearing completion, bar any extension work which needs writing up.

The progress report presentations fall during this period; clearly if the project is on track then there will be plenty to talk about. Since implementation should be nearing completion this is also a good point for a project review.

Milestones: All programming and implementation completed, leaving only testing, analysis and writing up left to complete. Progress Report Deadline - Fri 4 Feb 2011. Entire project reviewed both personally and with Overseers.

February 18th - March 11th

Complete any outstanding implementation work on possible extensions. Perform testing and obtain all results to be used in the analysis of the project. Ideally all testing should be complete, though again possible extension work may leave a small amount left to be done.

Milestones: Complete draft of the first three chapters (Introduction, Preparation and Implementation). Testing and analysis completed.

March 11th - March 18th (end of Lent term)

During this week the final two chapters (Evaluation and Conclusion) should be written up, completing a full draft of the dissertation.

Milestones: First complete draft of dissertation.

March 19th (start of Easter vacation) - March 26th

Review the entire dissertation. Insert any diagrams, graphs, tables and references which remain outstanding. Tweak advanced project presentation details such as formatting of code snippets. Focus on concision; remove any perceived wordiness and ensure project word count lies within the required range.

Milestones: Second complete draft, now ready for submission to DoS/supervisor.

March 27th - April 25th (end of Easter vacation)

During this 4 week period much time will be taken up by exam revision.

Submit the project to supervisors, DoS, fellow students and parents. Any feedback should be taken into account and the dissertation revised where necessary.

Milestones: By the end of the vacation have project complete and ready to submit.

April 25th - May 20th

This time should be left exclusively for exam revision. There should however, be just enough time to re-read the dissertation and make any final alterations, before final submission one week before the deadline.

Milestones: Submission of Dissertation - Friday 20th May. Date one week prior to deadline - Friday 13th May.