Firefox Enhancement Proposal:
Remote User Profiles

Andre Campos - 0230481 - drick@uvic.ca

Bryan Lane - 0434698 - blane@uvic.ca

Neal Clark - 0429078 - nclark@uvic.ca

Sunpreet Jassal - 0323709 - ssjassal@uvic.ca

Stephen Hitchner - 0430473 - hitchner@uvic.ca

1.0 Abstract

This report details a new feature proposal for Firefox 2. The proposed enhancement would allow users to export their user profile (all personal modifications including bookmarks, browsing history, installed plug-ins and extensions, cookies, address book entries, saved form data and saved passwords), or a portion thereof, to a central server, so that it could be downloaded at a new location. In this way, a user could have a consistent, customized web browsing experience no matter where he or she was browsing from, provided they were using Firefox.

Over the course of the report, the feature is first discussed, followed by two possible methods of implementing the feature (namely implementing the feature as a Firefox extension and implementing the feature by integrating into the existing Firefox core code base). After comparing these two methods in the final section, it was decided that the extension approach was the best option for implementing feature due to its lower development costs, lower maintenance costs, ease of deployment, and fewer unknowns.

2.0 Glossary

Extension - a small add-on that adds functionality to Firefox

Firefox - web browser created by the Mozilla Foundation
Flash - a multimedia authoring technology

HTML - Hyper Text Markup Language
HTTP - Hyper Text Transfer Protocol, a protocol used to transmit web pages.

Java Applet - a small Java application that can be run in a web browser

Mozilla - original public name of Mozilla Application Suite (aka SeaMonkey), created by the Mozilla Foundation

PKCS - Public Key Cryptography Standard
PKI - Public Key Infrastructure

SAAM - Software Architecture Analysis Method, a scenario-based method of evaluating software architectures
SQLite - a small C library that implements a self-contained, embeddable, zero-configuration SQL database engine.
SSL - Secure Socket Layer. A cryptography protocol

User Profile - a collection of data that is personalized to the user, including bookmarks, browsing history, cookies, plug-ins, extensions, address book entries, saved passwords and saved form data.

XDR - External Data Representation
XML - Extensible Markup Language
XPCOM - Cross Platform COM (component object model) implementation
XUL - XML User interface Language

3.0 Introduction

Many users of web browsers regularly access the World Wide Web from numerous machines from day to day. Users could perhaps log on at work, at home, on campus, maybe even an internet cafe. An inconvenience that these users have always had to deal with is that when one bookmarks a page when using one computer, that bookmark is not available on any other computer that the user may use to surf the web. In fact, many options that are available to help personalize the web browsing experience are not portable. From saved passwords to browsing history, from plug-ins to cookies, many users would love to be able to take their profile of features and settings with them, wherever they choose to browse from. Unfortunately, no such feature is available in popular web browsers today.

This paper will put forth a proposal to enhance Firefox with a new feature allowing users to modify and store their profiles on a central server which can then be accessed at a different location in order to keep the user's experience consistent across browsing locations.

4.0 Proposed Enhancement

The proposed feature aims to provide infrastructure for users to keep an updated copy of their profile in a centralized server such that whenever they use Firefox, from any computer, their profile will be loaded seamlessly. Moreover, users will be able to import/export their profile to a local file. The bottom line is that users will have the "same" Firefox no matter where they are.

4.1 Profile

In Firefox, similarly to Mozilla, a profile is a set of local files that represent the user identity. A profile has several components; however not all of them are pertinent to the proposed feature. The following components are in the scope of the *Remote Profiling* feature:

- Bookmarks: A list of the user's favorite pages that can be easily accessed through a convenient interface;
- Browser history: Record of which sites have been visited and in what order;
- Cookies: Parcels of text that are exchanged with a webserver such that stateful browsing is enabled.
- Plug-ins: Browser components that enables the browser to render certain types of content (i.e., Java Applets, Flash, etc)
- Extensions: Components that change the behavior of the web browser (i.e., this feature is an extension)
- Saved form data: Stored form data that the user has previously filled at some website. If the user is exposed to the same form again, the data will be filled automatically.
- Saved passwords: Securely stored passwords from websites that require authentication
- Address book entries: List of the user's contacts

4.2 Online Profile Repository

This feature is based on the fact that in order to use a web browser the user is connected to the internet. Consequently, an online profile repository will be kept such that users can upload their profiles in order to retrieve them when using Firefox in a different computer. The repository shall be able to keep versions of profiles, much like a source code repository. That way, users can rollback to determined version of the profile at their convenience.

4.3 Sub-features

This extension encompasses the following sub-features:

- Create online profile;
- Change password;
- Export profile to disk;
- Import profile from disk;
- Export profile to server;
- Import profile from server (latest version); and
- Import profile from server (any version).

This section elaborates each of the sub-features with brief use cases (main success scenarios only).

Use Case 1: Create Online Profile

User indicates he wants to create a profile. System shows form containing username, password and password confirmation. User inputs the appropriate data. System creates profile and stores login information for future use.

Use Case 2: Change Password

User indicates she wants to change her password. System shows form containing username, current password, new password, and new password confirmation. User inputs the appropriate data. System creates profile and stores login information for future use.

Use Case 3: Export Profile to Disk

User indicates they want to export the profile to disk. System prompts for which profile components the user would like to export (see Profile sub-section for a list). User selects profile components. System prompts user for location profile will be saved. User inputs location. System saves profile in one file and tells user that profile has been saved. System notifies user that Firefox needed to be restarted for changes to be applied.

Use Case 4: Import Profile from Disk

User indicates they want to import profile from disk. System prompts user for location of profile. User inputs location. System prompts user for which profile components user would like to import (see Profile sub-section for a list). User selects components. System loads profile (the current local profile is overwritten) and tell user that profile has been loaded. System notifies user that Firefox needed to be restarted for changes to be applied.

Use Case 5: Export Profile to Server

User indicates they want to export profile to repository. System prompts user for username and password. System prompts user for which profile components user would like to export. User selects components. System pushes profile into repository and tells user that profile has been saved in the repository. System notifies user that Firefox needed to be restarted for changes to be applied.

Use Case 6: Import Profile from Server (latest version)

User indicates they wants to import profile from repository. System prompts user for username and password. System prompts user for which profile components user would like to import. User selects components. System pulls profile from the server (the current local profile is overwritten) and tells user that profile has been loaded from the repository. System notifies user that Firefox needed to be restarted for changes to be applied.

Use Case 7: Import Profile from Server (any version)

User indicates they want to import and old profile from repository. System prompts shows a list of versions containing the dates they were updated. User selects desired profile. System prompts user for which profile components user would like to import. User selects components. System pulls profile from the server (the current local profile is overwritten) and tells user that profile has been loaded from the repository. System notifies user that Firefox needed to be restarted for changes to be applied.

5.0 Two Realization Strategies

5.1 Extension Approach

In this approach, we would provide functionality to have remote profiles by developing a extension (also called an add-on). This extension will give the user the ability to export his/her profile elements selectively; a user will be able to choose if he/she wishes to export any one or all of the following profile elements:

- bookmarks;
- browser history;
- cookies;
- plug-ins;
- extensions;
- saved form data;
- saved passwords;
- address book entries.

Once a user exports the profile elements to a file, he/she will be able to access it over HTTP. When the user opens Firefox on another computer, he/she would need to download our extension, which would enable the user to load the saved profile elements. Once again, the user will be able to choose which profile elements he/she wishes to import. Our extension will update the profile and restart Firefox permitting the user to access his profile elements on more than one computer.

However, this approach only permits using one browser instance with the loaded profile elements at a time to prevent data corruption arising from unsynchronized profile and browser.

5.3 Refactoring Details

Since this approach constitutes implementing an extension, the built-in extension mechanism in Firefox would be used to realize remote profile management.

This approach does not require any change to the architecture of Firefox; it would, instead, reuse the existing extensions support in Firefox to provide an extension to load and export profiles. In addition, this approach allows for ease of development and maintainence as the implementation can be done in high-level languages such as JavaScript and XUL, and no compilation would be required to test and deploy the extension.

5.4 Implications and Risks

System Reliability

Updates to the remote profile must be atomic. Only one user must be accessing his/her profile, and only one browser instance should be running at any given time as Mozilla Firefox needs to correctly update the profile. Reading the remote profile using multiple instances of Firefox may cause data corruption because the profile and the browser are not synchronized.

Security

Sensitive profile elements such as passwords and saved form data must be handled with care.

Maintainability

Our extension must be updated when the extension mechanism exposed by Firefox changes to prevent any version incompatibility issues.

Usability

Users must download our extension first before loading their saved profile. In order to update the default profile of Firefox, our extension will have to restart Firefox. Care must be taken to prompt the user about the restart and restore any open windows and tabs.

5.5 Firefox Codebase Approach

For this implementation approach the actual Firefox code base will be modified to add methods to directly talk to a profile repository. These methods would be added into the Data Persistence because the Data Persistence layer is currently responsible for maintaining the profile on disc.
As detailed in previous reports the Data Persistence layer is responsible for maintaining the following profile elements:

- bookmarks
- browser history
- cookies
- plug-ins
- extensions
- saved form data
- saved passwords
- address book entries.

This feature would add the ability for the use to save their profile to disc and/or to a remote profile repository. This feature would need to add the following functionality to the Firefox code base for this feature to work:

- common data format
- common data access functions
- network communication functions
- configuration UI

5.6 Common data format

Currently profile data is saved in several different data formats. In order to backup the profile to a remote site a common data format will need to be developed to represent the profile data. Since Firefox is already very good at parsing XML, it makes sense that the new data format be XML based. Although verbose, all the data contained in a Profile can be easily presented in some kind of XML format.

5.7 Common data access functions

Currently the data persistence layer does not contain a uniform method for data storage. Future versions of Firefox intend to more towards
having a consistent SQLite backend, but currently this is not the case. Currently there are method that convert the internal data structures into a format that serializable to disc. Additional methods need to be added to allow the new feature to serialize the internal data structures into our XML data format.

5.8 Network Communication Functions

Firefox alread provides an extensive array of network access functions. This feature can leverage off the existing network function for storing and retrieving profiles from remote profile repositories.

5.9 Configuration UI

A configuration UI would also need to be added to allow the user to configure their remote profile repository. This UI would be written in the XUL language which would be easily added to the existing UI layer. Additional information would need to be saved by the Data Persistence layer to record the necessary information required for the user to access their remote profiles.

5.10 Refactoring Details

The Data Persistence layer would require major refactoring to implement this feature.

5.11 Comparison of Implementation Options

The first option for implementing the desired remote profile functionality in Firefox is to modify Firefox's code-base directly and tap directly into the various components. This approach would provide the greatest control and flexibility when it comes to implementation. Such an implementation could have better performance and possible improved security over other implementations. This approach also has a number of drawbacks including increased development costs as more in depth modifications are necessary to modify and compile a customized version of the software. Such modification would be nontrivial as a number of different modules would need to be modified in order to connect the new functionality. Additionally maintenance costs would be much higher since the modification are unlikely to make it into the official distribution or Firefox and therefore whenever a security update or new version of Firefox is released the new module would need to be ported over.

The second option for implementation would be to take advantage of the Firefox's extension capabilities to implement the desired functionality using a combination of JavaScript, RDF files, and XUL Interfaces. This approach does not require source code modification for its implementation but instead utilizes existing functionality to extend the browsers capabilities. Firefox has existing functionality to facilitate automatic updating of extensions and also provides support for migrating extensions to newer versions of the browser. This approach has a number of unknowns which could pose a problem to this implementation including access to data. In order to store a users profile remotely the extension must have access to sensitive information such as passwords and cookies which Firefox may block access to for security reasons.

6.0 Conclusion

It was decided that the proposed remote profile functionality would be implemented as a Firefox extension because of the lower development and maintenance costs, fewer unknowns, and existing automatic update functionality. A Firefox extension would have substantially lower deployment costs since it would require only a few files to be delivered to the client instead of a replacement browser. It is also much easier to download and Install a Firefox extension than it would be to download and install a complete web browser. In order for the remote profile functionality to be useful it must be easy to install the necessary functionality on as many Firefox installations as possible. It is clear that a Firefox extension would provide the greatest benefits to the customer with the fewest risks.

7.0 References

[1] http://web.uvic.ca/~hitchner/concrete.pdf

[2] http://www.csc.uvic.ca/~vlahod/a2.pdf

[3] http://www.mozilla.org/support/firefox/

[4] http://www.sqlite.org/