Web Browser Architecture Proposal with Local Agent

Laxman A. Rasal
Department of Computer Engineering and Information Technology,
College of Engineering Pune,
Pune - 411005, Maharashtra, India
E-mail: rasalla07@comp.coep.org.in

V. Z. Attar
Department of Computer Engineering and Information Technology,
College of Engineering Pune,
Pune - 411005, Maharashtra, India
E-mail: vahida@comp.coep.org.in

Abstract - The web browser has become the centerfold application to the Internet. It has evolved from being an application for rendering static web pages to a host of a variety of applications. The growing requirements for a web browser have lead to code additions, vendor specific add-ons but not much change in its base design.

Hence in this paper we are proposing new web browser architecture based on local computing which will ensure improvement in Quality of Service (QOS). This can be achieved by adding a generic extensible local agent on client end which will interact with web browser. This new local agent will support some generic functionality like compression. Additionally this agent can also have an additional security layer specifically designed for Add-ons, authentication, data upload and download.

By separating out specific security features from the web browser to a separate component will help to reduce the code size of a web browser. Security patch-ups for newly found vulnerabilities related to web-browser can be made available easily without the reinstallation of complete web browser. In case of multimedia data upload, compression will ensure tremendous reduction in data transfer over network. In case of image uploads using compression; our POC proves the best case reduction in data transfer by 80 times and average case data transfer reduction by 12 times

Keywords: web browser, local computing, distributed computing, compression, QOS.

I. INTRODUCTION

A web browser is a program that retrieves documents from remote servers and displays them on screen, either within the browser window itself or by passing the document to an external helper application. It allows particular resources to be requested explicitly by URI, or implicitly by following embedded hyperlinks.

The World Wide Web (WWW) was first described in a proposal written by Tim Berners-Lee in 1990 at the European Nuclear Research Center (CERN) (Berners-Lee, 1999) [1]. By 1991, he had written the first web browser, which was graphical and also served as an HTML editor.

The web browser has become the centerfold application to the Internet. It has evolved from being an application for rendering static web pages to a host of a variety of applications, where each web page represents a distinct application, such as a news feed, an email client or a video application. The growing requirement for a web browser has lead to code additions, without much change in its base design. Add-ons and plug-ins are coupled with the web browser to accommodate vendor specific functionalities for web applications or services. Web browsers are dependent on third-party firewalls and anti-virus solutions for ensuring security of the host. This has caused easy spread of botnets. The growing size of the browser code and reluctance to change the design of the browser has made it a welcome door for number of exploits.

The growing popularity and usability of web is enforcing to redesign the web browser architecture. The new web browser architecture should be able to overcome the existing vulnerabilities of web browser as well as to make it suitable and sustainable for future web applications. The future applications will involve bandwidth hogging multimedia applications, real time critical applications as well as scientific distributed computing applications developed using cluster of computing nodes across the internet. We also need to ensure that we are able to reduce the huge data traffic getting generated in backbone network due to new popular web applications such as social networking web-sites, image and video sharing web sites.

II. EXISTING ARCHITECTURE OF WEB BROWSER

A. Overview

The World Wide Web (WWW) is a universal information space operating on top of the Internet. Each resource on the web is identified by a unique Uniform Resource Identifier (URI). Resources can take many different forms, including documents, images, sound clips, or video clips. Documents are typically written using Hypertext Markup Language

978-1-4244-4711-4/09/$25.00 ©2009 IEEE

(HTML), which allows the author to embed hypertext links to other documents or to different places in the same document. Data is typically transmitted via Hypertext Transfer Protocol (HTTP), a stateless and anonymous means of information exchange. A web browser is used as a client application to locate and display the information available on World Wide Web.

The exiting architecture of a web browser is as shown in Figure 1 [1]. It comprises eight major subsystems plus the dependencies between them:

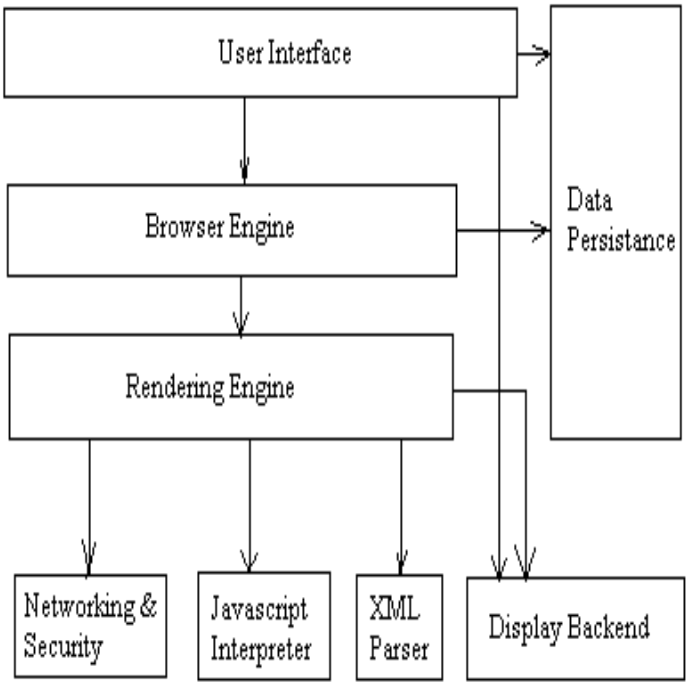

Figure 1: Architecture of Web Browser

- The User Interface subsystem is the layer between the user and the Browser Engine. It provides features such as toolbars, visual page-load progress, smart download handling, preferences, and printing.
- The Browser Engine subsystem is an embeddable component that provides a high-level interface to the Rendering Engine. It loads a given URI and supports primitive browsing actions such as forward, back, and reload. It provides hooks for viewing various aspects of the browsing session such as current page load progress and JavaScript alerts.
- The Rendering Engine subsystem produces a visual representation for a given URI. It is capable of displaying HTML and Extensible Markup Language (XML) documents, optionally styled with CSS, as well as embedded content such as images. It calculates the exact page layout and may use "reflow" algorithms to incrementally adjust the position of elements on the page. This subsystem also includes the HTML parser.
- The Networking and Security subsystem implements file transfer protocols such as HTTP and FTP. It may implement a cache of recently retrieved resources. It also imposes the security constraints as per the configuration made.
- The JavaScript Interpreter evaluates JavaScript (also known as ECMAScript) code, which may be embedded in web pages. JavaScript is an object-oriented scripting language developed by Netscape. Certain Java- Script functionality, such as the opening of pop-up windows, may be disabled by the Browser Engine or Rendering Engine for security purposes.
- The XML Parser subsystem parses XML documents into a Document Object Model (DOM) tree. This is one of the most reusable subsystems in the architecture. In fact, almost all browser implementations leverage an existing XML Parser rather than creating their own from scratch.
- The Display Backend subsystem provides drawing and windowing primitives, a set of user interface widgets, and a set of fonts. It may be tied closely with the operating system.
- The Data Persistence subsystem stores various data associated with the browsing session on disk. This may be high-level data such as bookmarks or toolbar settings, or it may be low-level data such as cookies, security certificates, or cache.

The web browser interacts with remote web services to locate and retrieve information made available on www. After validating the security checks it retrieves the web data made available and displays it to the user in browser window. To avoid network delays it also caches some information on client end so as to avoid round trip delay. The web browser has generic security and privacy policies. Hence deploying third party security applications is always recommended and mandatory as well.

B. Existing Bottlenecks

The current web browsers run as single application with minimum or no isolation between various browser components such as rendering engine, script engine, etc. The inclusion of all basic services in the browser has three big drawbacks: increased code size, lack of flexibility and bad maintainability. Bug-fixing or the addition of new features

means restarting the whole browser. An issue in one browser component or extension leads to a complete browser crash.

The web browsers have generic security policies which fall short in securing the host. This mandates the user to purchase and deploy third party security solutions. Since web browsers are getting utilized by malicious users to spread viruses, worms, spy wares and malicious add-ons; security components necessary to defend against these anomalies must be coupled with web browsers without extra additional cost. The growing size of the browser code and the reluctance to change the design of the browser has left the browser open to a number of exploits. According to a recent report [4], Symantec documented 93 vulnerabilities in Internet Explorer, 74 in Mozilla browsers, 29 in Safari, and 11 in Opera. In addition to these browser vulnerabilities, Symantec also documented 301 bugs in browser plug-ins over the same period of time.

Present browsers do not support data compression during data collection from client machine. Looking at growth in web based multimedia applications; early consideration needs to be given to deal with future growth and requirements of these advanced applications.

Nowadays even common client machines are becoming more powerful in terms of processing power. So in future, web applications will and should get developed which emphasizes on utilizing local computing power rather than assigning almost all computation to server. Similarly distributed computing applications will also prove to be more essential and economical than supercomputing. Hence future web-browser should be capable of supporting local and distributed computing without compromising security and privacy of the host.

III. PROPOSED WEB BROWSER ARCHITECTURE

The current web browsers run as single application with minimum or no isolation between various browser components such as rendering engine, script engine, etc. After studying the bottlenecks experienced in existing browser design (2.2); it necessitates to redesign the web browsers to take over the limitations faced.

In response to overcome the bottlenecks faced as well as to respond to future needs of web applications we are making an attempt to propose new web browser architecture. In the new architecture we are proposing to add a “Local Agent”. This agent will comprise of two main components “Security Manager” and “Computing Component”. The rest all components of web browser are maintained as it is with only addition of new isolated local agent. This agent will be a local component interacting with the web browser which will be able to utilize local computing power for various computation extensive features.

The proposed new architecture of a web browser is as shown in Figure 2. It comprises of all eight existing components plus an additional “Local Agent” comprising of two components.

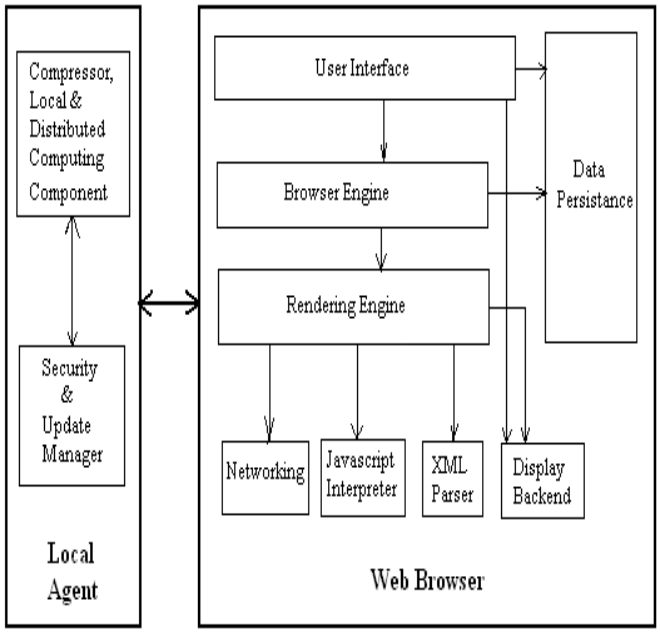

Figure 2: Proposed Web Browser Architecture

A. Local computing and Security Layer:

The first component, “Security & Update Manager” will be responsible for securing host during communication of host with internet applications. The update manager integrated with security manager will insure live update of security policies for newly found threats. Thus security and update manager will help protect host from all types of downloads and installations taking place online with latest update of all newly observed threats.

The second component, “Compressor, Local & Distributed Computing Component” will have computational extensive APIs made available to Web Applications. The APIs for compression will help web-sites to upload client end files on server in compressed format. This will ensure relatively less data flow and hence fast response time. The compression feature will help to reduce tremendous reduction in traffic on backbone network in case of multimedia data transfer from host to server. Similarly user can have personalized secure communication sessions by offloading the overhead of mutual authentication and encryption during data transfer to this local computing subsystem. Latest web sites

developed are extensively using AJAX feature to avoid server round-trips and hence to achieve fast response time with reduced data transfer. Likewise the local computing feature made available through this component will help future web applications to utilize local computing power offloading the servers for other computation tasks. The compression performed on client end is a good example of local computing. The third most important feature expects implementation of a standard framework supporting distributed computing. This will involve support for task distribution, synchronization, consistency and fail-over mechanism. The distributed computing feature will help development of future distributed web applications overriding the economical barrier of super-computing.

B. Proposed Implementation and experimental results

The new component will be closely tied-up with web browser. The four major features of that subsystem should be as follows:

- Independent: The local agent should be a separate component interacting with web browser.
- Configurable: User should be able to configure this component independently as per his needs. This component should have a live update feature which will ensure easy and immediately responsive web browser to new threats.
- Compression: The compression feature will be helpful to reduce the data transfer over backbone network of web. This feature will help to reduce tremendous data flow during multimedia data collection from end user. We have performed a proof of concept (POC) for image file getting uploaded on social networking & image sharing web sites; using JPEG compression and image resize. The following two tables shows the results obtained:

Type of Image (Resolution 640x480)	Original Size (In KB)	File size after compression (In KB)	With JPEG compression; % file size reduction	Conventionally: 'X' times data reduction
TIF	1100	67.1	1539	15
BMP	900	67.1	1241	12
PNG	768	67.1	1045	10
GIF	132	108	22	0.2
JPEG	67.1	67	0	0

Table 1: JPEG compression results

Change of Resolution & image size reduction	Size (In KB)	File size after resizing (In KB)	% file size reduction	Conventionally: 'X' times data reduction
JPEG (2000x1500)	450	53	749	7
JPEG (640x480)	53	53	0	0

Table 2: Image resizing results

Our POC proves the best case compression reduces data transfer by 80 times and average case data transfer reduction by 12 times.

- Local & Distributed Computing: A new standard framework should be made available to support local and distributed computing of future web-applications. This ensures best usage of local computing power easily available today due to huge reduction in cost of hardware resources. At the same time this causes reduction in backbone network traffic.

IV. RELATED WORK

Recently a new browser design was proposed co-relating the functionality of web browser to operating system," On the Design of a Web Browser: Lessons learned from Operating Systems" [2].

The paper analyzes requirements of an operating system (OS) design from the prospect of designing a new web browser. They study the characteristics of a micro-kernel based OS to determine how they can be perceived in a web browser, and propose a new browser design that attempts to overcome the shortcomings of current web browsers.

V. CONCLUSION

The proposed change in architecture for web browser ensures effective usage of the computing power available on client machines which is easily available today. By relieving the web browser from few privacy and security features helps to reduce the size of web browsers. This will also ensure easy maintainability of web browsers by independently tying-up live security update feature with the browser. The compression feature of the newly added component will certainly help to improve the QOS requirements of new multimedia based web applications. The proposed change in web browser architecture advocates the usage of local computing power in co-ordination with more stringent and specific security features.

VI. REFERENCES

[1] Alan Grosskurth, Michael W. Godfrey, “Architecture and evolution of the modern web browser”, http://grosskurth.ca/papers/browser-archevol-20060619.pdf

[2] "On the Design of a Web Browser: Lessons learned from Operating Systems", http://w2spconf.com/2008/papers/s2p2.pdf

[3] W3C Working Draft 8 August 2003 “Web Services Architecture” , http://www.w3.org/TR/2003/WD-ws-arch-20030808/wsa.pdf

[4] D. Turner. Symantec internet security threat report Technical report, Symantec, Sept. 2007