Browser as a Service (BaaS): Security and Performance Enhancements for the Rich Web

Nils Gruschka[1] **and Luigi Lo Iacono**[2]

1 **NEC Laboratories Europe**
Heidelberg, Germany
`nils.gruschka@neclab.eu`

2 **European University of Applied Sciences (EUFH)**
Brühl, Germany
`l.lo_iacono@eufh.de`

Abstract

This paper introduces an architectural approach to access the Web via a virtual Web browser executed within a secure Cloud environment.

Keywords and phrases Web, Security, Performance, Cloud, SaaS

Digital Object Identifier 10.4230/OASIcs.KiVS.2011.208

1 Introduction

The Web has become an indispensable prerequisite of everyday live and the Web browser is the most used application on a variety of distinct devices. The content delivered by the Web has changed drastically from static pages to media-rich and interactive Web applications offering nearly the same functionality as native applications, a trend which is further pushed by the Cloud and more specifically the Cloud's SaaS layer. In the light of this development, security and performance of Web browsing has become a crucial issue.

From a security perspective, Web browsers superseded other attack targets like email [1] and are more and more the gate for installing malware on victims computer [2]. No matter if malicious or reputable Web sites, both may contain manipulated active content. In the most dangerous form—the so-called drive-by downloads—the malware is downloaded and installed to the visiting client without knowledge and notice of the user.

Another problem of Web browsing is the "memory" of the browser. Browser cookies and histories can lead to privacy breaches and especially becomes a problem if recognition can be performed between different Web sites. A well known attack of this type is called *history stealing* allowing a Web site to access the user's history using JavaScript and CSS [3]. In an advanced form this attack even allows to identify a user—i.e. learning his name, address etc.—by analyzing the sites a user has visited [5]. And even if a user disables or cleans his cookies and browser history, recent published techniques are still able to store re-identification data [4].

Further, performance issues arise from the increased complexity of Web applications running inside the browser. Nearly all functionalities of native applications can nowadays be found in the browser part of Web applications. These programs—most commonly written in JavaScript—require high memory and processing resources. This makes JavaScript engines and specifically their efficiency the most promoted feature for new browsers or browser versions. In addition to JavaScript also CSS rendering or DOM access are resource consuming and can become an execution bottleneck [6].

This paper contributes and discusses an architectural approach towards faster and more secure Web browsing taking into account contemporary attack vectors and adhering to the demands of the SaaS era.

2 Browser as a Service

The basic underlying idea of the proposed architecture is to provide access to a remote Browser as a Service (BaaS, see Figure 1). The remote browser is executed in a secure cloud environment for a very limited lifetime. The interaction with the remote browser takes place with a locally installed browser. In contrast to the current "standard" scenario, the locally installed browser does not execute any active content, besides the one required to capture user inputs and events as well as to display the graphical output the remote browser has rendered and sent.

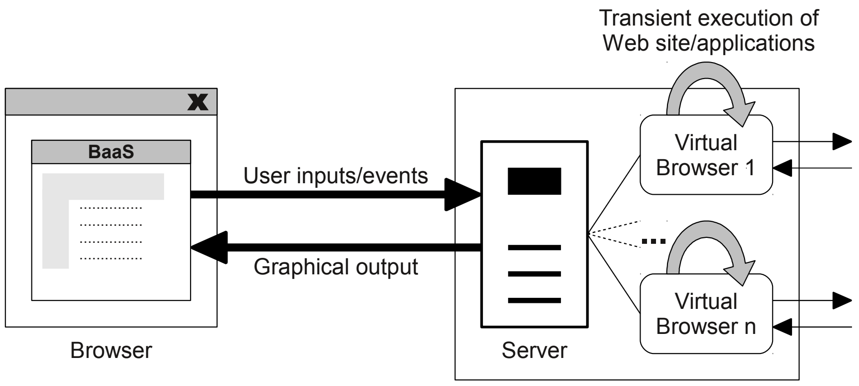

Figure 1 BaaS Architecture

The server-side runs a virtual machine containing a Web browser instance denoted as *Virtual Browser*. Such an instance runs an up-to-date Web browser engine which includes all state-of-the-art extensions required to consume services and applications in the Web. The lifetime of the instance is transient. It does not save any state and is only available during the actual session of the user. Henceforth, any malicious code eventually installed to the virtual browser while surfing the Web with this particular instance will be destroyed as well.

The client-side still requires a Web browser to access and use a virtual browser. The virtual browser appears within the local browser's output area as a Web application (browser in a browser). When the user interacts with the virtual browser within the local browser, the inputs and events are captured by the local browser and then send to the virtual browser. It then triggers the related actions and renders the associated output accordingly. This includes the execution of active content which is performed exclusively within an isolated environment on the remote server. Thus, a possible infection will harm the proxying virtual browser instance, but not the acting end-user client, as the virtual browser does only transmit the rendered output to the client.

Additionally, such an approach has positive effects on the performance of executing the client-side portion of Web applications. Since this part is increasing steadily—especially in SaaS-based applications—the demands in terms of local computing resources are increasing in the same way. This is mainly caused by the trend to offer Web applications with a similar user-experience like in standalone programs. The proposed BaaS approach off-loads the execution of complex and large JavaScript programs to the virtual browser, which enables the usage of resource-demanding Web applications also for restricted devices such as thin clients, netbooks and smart phones.

3 Conclusion and Open Issues

Performance of complex Web applications, privacy breaches and malware vulnerabilities are major problems of today's Web usage. The presented idea for a virtual browser running "in the cloud" has the potential to provide a solution for these issues. However, a number of issues and questions have to be solved for realizing BaaS. The main problem is the protocol for accessing the virtual browser. Standard remote desktop protocol (like RDP or RFB) might be sufficient for office applications or online shopping but probably not for video applications. In any case, parameters like responsiveness, rendering quality and network traffic must be studied intensively.

References

1 David Barroso. Botnets – the silent threat. *ENISA Position Paper*, 2007.

2 Marc Fossi, Dean Turner, Eric Johnson, Trevor Mack, Téo Adams, Joseph Blackbird, Stephen Entwisle, Brent Graveland, David McKinney, Joanne Mulcahy, and Candid Wueest. Symantec global internet security threat report: Trends for 2009. XV, April 2010.

3 Artur Janc and Lukasz Olejnik. Feasibility and real-world implications of web browser history detection. In *W2SP 2010: Web 2.0 Security & Privacy*, 2010.

4 Samy Kamkar. evercookie – never forget. `http://samy.pl/evercookie/`, 2010.

5 Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to de-anonymize social network users. In *IEEE Symposium on Security and Privacy*, 2010.

6 Kaimin Zhang, Lu Wang, Aimin Pan, and Bin Benjamin Zhu. Smart caching for web browsers. In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 491–500, New York, NY, USA, 2010. ACM.