Security Evolution of the Webkit Browser Engine

Renáta Hodován  
Department of Software Engineering  
University of Szeged  
Dugonics tér 13., 6720 Szeged, Hungary  
hodovan@inf.u-szeged.hu

Ákos Kiss∗  
Department of Software Engineering  
University of Szeged  
Dugonics tér 13., 6720 Szeged, Hungary  
akiss@inf.u-szeged.hu

Abstract—By now, the Web has matured into a full-scale application platform and its popularity is constantly rising. However, popularity comes at a cost: the Web is becoming more and more of a target for attackers. In this paper, we argue that the security analysis of web browsers deserves attention and the data on their security evolution is of high value. We examine the security evolution of the widely used WebKit browser engine, investigate its historical security data, and point at the alarming trends.

Keywords—Security; web browser; evolution

I. INTRODUCTION

By now, the Web has matured from a distributed document storage system, what it was in the beginning, into a full-scale application platform. Its popularity as a platform is constantly rising [1] and is comparable to the desktop domain. This is all due to scripting, i.e., the extension of both the server-side and the client-side with code execution capability. Scripting allowed the generation of dynamic content and smooth user interaction. Today, the “web programming languages” like JavaScript and PHP are among the most popular and widely used languages [2]. However, such popularity comes at a cost: the Web is becoming more and more of a target for malicious attackers. Thus, its security is of major concern. In fact, since the mid-2005, more security vulnerabilities have been reported for the Web than for the desktop domain [3].

When the Web and its security is discussed, focus mostly falls on the security of web applications [4], or the applications and the browsers are handled as one joint category [3] (unlike in the desktop domain, where the applications and the operating system are discussed separately). However, similarly to the web applications that are the analogous counterparts of user-space desktop applications, browsers and browser engines correspond to operating systems and the kernel [5]. Thus, we argue that the security of web browsers deserves to be analysed separately as well.

In this paper, we take a look at the security evolution of the widely used web browser engine, WebKit [6]. We investigate historical security data and analyse the trends.

The rest of the paper is organised as follows: In Section II, we introduce the WebKit project and the trends of its development. In Section III, we analyse the security evolution of the browser engine by mining several databases. In Section IV, we give a brief overview of the related publications. Finally, in Section V, we summarize the paper and outline our plans for future work.

II. DEVELOPMENT TRENDS

We have chosen WebKit [6] as the target of our analysis since it is the most widespread browser engine today. It powers several desktop browsers (Apple Safari and Google Chrome among others) and mobiles as well (like those shipped with iPhone, Android, and MeeGo phones). According to the April 2012 statistics of StatCounter [7], WebKit holds more than 38% of the browser market share.

Although not all of the above mentioned browsers are developed in the public, their core, the WebKit browser engine is an open source project. Thus, we can access its code and bug repositories, and analyse the rate of its development. Figures 1 and 2 show ever-increasing trends. At the end of 2007, the size of code in WebKit was cca. 680 KLOC (thousand lines of code), while by April 2012, it became almost 2.3 MLOC, which is a more than 3-fold increase. Not only is the code constantly growing but it is changing faster and faster. In March 2012, the revision number of the code repository was incremented by more than 3000, which means that the code was modified in every 15 minutes on average.

III. SECURITY EVOLUTION

One might naively assume that the quality (and thus, the security) of software which has been developed for a long time – and the development of WebKit started more than
10 years ago – is improving over time. However, Pressman forecasts that continuous changes and extensions are likely
to introduce undesired behaviour, i.e., defects to the code [8].
Thus, our hypothesis is that such an increase in size and such
a pace of development as observable in WebKit might have
an adverse effect on the security of the project in the long
run.
To verify our hypothesis, we have started to analyse the
publicly available sources of information on the security
problems of WebKit. First, we focused on the repositories of
the project itself. The bug database of WebKit has a special
section for issues that are labelled as security-related by
the experts of the developer community. By relating the code
repository and the bug database, we have been able to collect
the changesets that belong to security bugs. When building
our security database, we also recorded the date of the
modifications and the size of the changesets. Altogether, we
have found 877 security bug entries and 998 modifications
(with some bug fixes spanning over multiple changesets).
Our findings, which are depicted on Figures 3 and 4,
underpin our hypothesis. Both the number of the modifications
fixing security problems and the size of the code affected
by these modifications are increasing. In 2007, only 0.38%
of the changes were security-related, but in the first four
months of 2012, this number crept up to 2.03%, which, as
an absolute value, is not unacceptably high but shows a more
than 5-fold increase in the efforts spent on fixing security
problems.
Additionally to the project repositories, we have mined the
Common Vulnerabilities and Exposures (CVE) List [9] for
WebKit-related entries. We have found 356 reported vulnera-
bilities between the years 2007 and 2011. The description of
the CVEs allowed us to categorize them manually according
to their severity. For rating the vulnerabilities, we have used
the guidelines of the Chromium project [10]. The trends are
shown in Figure 5. The gross numbers are in line with the
trends observed in the code repository and are shown in
Figure 3. The number of CVEs are rising almost every year
(with 2011 being a bit more “fortunate”). Most importantly
(or distressingly), the ratio of critical vulnerabilities is rising
as well: in 2007, only 19% of the vulnerabilities were rated
critical, but by showing growing trends, this ratio rose to
84% in 2011.

IV. RELATED WORK
From the dawn of the internet era, when it was only
used by the military and by scientists, to our days when
we are managing even the paying of bills via ebank, the
Internet has been working with sensitive data. This calls
for the need to determine the rules and the restrictions to
reduce abuses. The first articles in the web security topic
were written at the end of the ‘80s. These first papers
focused primarily on the security of data transmission [11].
Recently however, as the Web is becoming dynamic and
the number of web applications is continuously increasing,
interest has turned to the security issues of web applications.
Experts have been trying to draw up guides on writing secure
V. SUMMARY AND FUTURE WORK

In this paper, we have argued that the security analysis of web browsers deserves attention and the historical data on their security evolution is of high value. We have analysed the widely-used WebKit browser engine from the perspective of security and found that the trends are alarming. Based on the data extracted from two data sources, we have found that more and more security problems are arising and their fixing is requiring increasing efforts. Additionally, the ratio of the critical bugs is also rising.

Our ultimate goal is not only to observe the trends of browser security but also to help developers avoid the introduction of security holes or to help them recognise the problems at an early stage. We plan to thoroughly analyse the patch database we have extracted from the code repository of WebKit and apply code analysis techniques (e.g., computation of metrics [15], [16]) on them or on the affected parts of the code to find characteristic common attributes. We hope that such characteristics can help us point out bad smells, suspicious parts of the code.

Additionally, we are also interested in defining an attack surface metric [17] for web browsers (which is ultimately also a good indicator of the security of a product). The information gathered in this work will help to validate the defined metric.

REFERENCES


