BURPKIT
Using WebKit to Own the Web

Presented by:
Nadeem Douba

2015-07-15
INTRODUCTION

• **Nadeem Douba**
  • Founder of Red Canari, Inc.
  • Based out of Ottawa, ON.
  • Hacker

• **Interests:**
  • Exploiting stuff
  • Building hacking tools

• **Prior work:**
  • Sploitego (presented at DEF CON XX)
  • Canari (used by Fortune 100s)
  • PyMiProxy (used by Internet Archive)
OVERVIEW

• WebKit
  • What is it?
  • Why use it?
  • How can we use it?

• BurpKit
  • Design Considerations
  • Implementation
  • Demos!

• Conclusion

• Questions?
THE WEB PEN-TESTER’S CONUNDRUM

• Today’s web applications are complex beasts
• **Heavy** use of JavaScript for:
  • Rendering pages
  • Rendering page elements
  • Performing web service requests
• ¿But our **security tools** are **still** scraping HTML!?
OUR TOOLKIT

• **Reconnaissance & Scanning:**
  • Most tools (nikto, cewl, etc.) just scrape HTML

• **Attack:**
  • **BurpSuite Pro/Community**
    • Lobo-based Renderer tab (Burp’s neglected child)
  • No JavaScript/HTML5 support
  • Charles & Zed are just proxies
  • WebSecurify’s Proxy.app only has a web view
MODERN TOOLKIT REQUIREMENTS

• Web penetration testing tools that:
  • Have modern web browser capabilities
  • Parse and interpret JavaScript
  • Dynamically render and inspect content

• **Most importantly:**
  • Our tools need to be able to interact with the DOM!
WEBKIT

What is it good for? - Lots of things!
“WebKit is a layout engine software component for rendering web pages in web browsers. It powers Apple's Safari web browser, and a fork of the project is used by Google's Chrome web browser.”

(UN)OFFICIAL DEFINITION...

WebKit is basically a collection of use-after-frees that somehow manages to render HTML (probably via a buffer overflow in WebGL).

9:34 PM - 29 Apr 2014

Follow

the grugg
@thegrugg
WEBKIT API

• Made up of two major components.

• JavaScriptCore - responsible for everything JavaScript:
  • JavaScript/JSON parsing & execution
  • Garbage collection
  • Debugger
  • Etc.

• WebCore – responsible for everything else:
  • Resource loading
  • Content parsing & rendering
  • Web Inspector
  • Etc.
KNOWN IMPLEMENTATIONS & FORKS

- Apple’s Safari
- Android’s web browser
- Nokia QT
- JavaFX WebView
- WebKitGTK+
- PhantomJS
- Google Chromium
- Node WebKit
- Many more…

(*https://trac.webkit.org/wiki/Applications%20using%20WebKit*)
WHY USE WEBKIT?

**Pros**

- Widespread adoption
- Lots of language support: Java, Python, C/C++, JavaScript, etc.
- Portable across many platforms
- Can interact with the DOM and JS Engine.

**Cons**

- Your code will be susceptible to the same bugs that plague modern browsers
- Tools will be hungrier for system resources (i.e. RAM, CPU).
HOW CAN YOU USE WEBKIT?

**Language**
- JavaScript (NodeJS)
- Python
- JAVA
- Swift/ObjC
- Ruby
- C/C++

**Libraries**
- Node WebKit
- WebKitGTK+, PyQt
- FX WebView, Qt Jambi, JxBrowser
- UIWebView
- WebKitGTK+, Qt
- Chromium, WebKit
How we used WebKit.
**WHAT IS BURPKIT?**

- **BurpKit** = **BurpSuite** + **WebKit**
- Used JavaFX’s implementation of WebKit
  - WebView & Debugger
  - WebEngine
- Provides a *real* rendering tab (that’s right... no more lobo)
- Has a bidirectional bridge between BurpSuite & WebKit!
- And more!
DESIGN DECISIONS

• Chose to go with JavaFX over JxBrowser – why?

• Redistribution:
  • JavaFX comes with Java 1.8+.
  • JxBrowser needs bundling (>250MB)

• Cost:
  • JavaFX is FREE!
  • JxBrowser is not!

• API:
  • JavaFX has a cleaner API
  • JxBrowser’s is a bit ¿clunky?
JAVAFX: PROS AND CONS

Pros

- Easy-to-use & clean API
- Complete JavaScript bridge
- Portable across many platforms
- Leverages the Java URL framework (hookable)
- Does provide debugging/profiling information (with some hacking)
- Bundled with Java 1.8+

Cons

- API is incomplete – under development
- No GUI components for WebInspector and friends
- Little documentation on advanced features (must look at code)
- Still a bit buggy
IMPLEMENTATION

Nerd Rage
CHALLENGES

• Burp uses **Swing** for its GUI
  • WebView and WebEngine need to run on **FX** event loop

• WebEngine does not have a `loadContentWithBaseUrl(content, url)` method - only has:
  • `loadContent(content, type);` and
  • `load(url)`

• **BurpSuite** had to be able to interact with **JavaScript** and vice-versa
CHALLENGE: SWING/FX INTEROP

• Solution: javafx.embed.swing.JFXPanel

• Gotchas:
  • Must avoid interweaving blocking calls
    • i.e. Swing → JavaFX → Swing = ¡DEADLOCK!
  • Always check if you’re on the right event loop!

• Workarounds:
  • Eagerly initializing resources sometimes necessary
  • Lots of wrapping code!
**CHALLENGE:** LOADING CONTENT WITH A BASE URL

- **Why?**
  - Required to render responses for repeated requests

- **Solution:** hook `java.net.URL` protocol handling framework
  - WebView uses framework to issue HTTP(S) requests

- **Challenge:**
  - Our new handlers would have to support both live and repeated requests.

Credit: http://media.techtarget.com/tss/static/articles/content/dm_protocolHandlers/java_protocol.pdf
**CHALLENGE: REPEATER**

- **Background:** did not want to reissue a live request because content may change.
- **Solution:** overrode HTTP(s) handlers and used User-Agent to “tag” repeated requests.
  - If User-Agent contains SHA1 hash, give URL handler fake output stream
  - Else, continue with live request
- See BurpKit Java package `com.redcanari.net.http` for code.
**CHALLENGE:** JAVASCRIPT BRIDGE

- **Background:** need to be able to query and manipulate DOM
- **Solution:** inject JAVA objects into JS engine!
- **Gotchas:**
  - Funky reflection algorithm in WebEngine prevented straight-forward JAVA object interaction.
  - Lots of deadlock scenarios
- **Workarounds:**
  - Wrapper classes galore!
  - Eager instantiation of Swing components.
Google: Before

& After
"when something exceeds your ability to understand how it works, it sort of becomes magical."

Jonathan Ive
BURPKIT DEMOS

There’s lots to see!
DEMO: GUI WALKTHROUGH

Feature set
XSS TRACKER

Tainting applications
DEMO: DOM INTERACTION

Analyzing Twitter Followers
DEMO: BURP EXTENSIONS

Proxy Listeners, Message Editors, and Context Menus
CONCLUSION

• Let’s stop scraping and let’s start DOMinating the web!

• Our security tools need to evolve just like the web.
  • We have the tools/libraries at our disposal

• Please contribute your ideas and code to BurpKit!
  • We need to make it the standard!
• My *Lovely* Wife 😊
• Justin Seitz
  • [http://automatingosint.com/](http://automatingosint.com/)
• Dirk Lemmermann
  • [http://dlsc.com/](http://dlsc.com/)
• Tomas Mikula
• Java/JavaFX team
• The Noun Project
• All the contributors!
¿QUESTIONS?
We aim to please…