

Browser Rootkits

Christophe Devaux - christophe.devaux@sogeti.com
Julien Lenoir - julien.lenoir@sogeti.com

Sogeti ESEC R&D



Agenda

- 1 Introduction
- 2 Rootkit for Firefox
- 3 Rootkit for Internet Explorer



Introduction

Why design a web browser rootkit?

Today's browsers

- Browsers are getting so complex that they can be considered as operating systems
- Browsers are usually allowed to access the Internet

Constraints

- Be as furtive as we can
- Be exploitable with user rights only



FIREFOX



Content

- 1 Introduction
- 2 Rootkit for Firefox
 - One add-on to rule them all
 - Hide the devil inside
 - Communication and Spreading
 - Payloads
 - Conclusion
- 3 Rootkit for Internet Explorer



Main principles

Build a Firefox add-on like a traditional rootkit kernel module

Attributes:

- Loads and becomes persistent
- Hides itself (from the browser scope)
- Communicates and answers to orders

Constraints:

- Exploitation with minimal user rights
- Focus on the stealth of the solution
- Multiplatforms



What is an extension?

An extension...

- is a simple compressed file with JavaScript/XUL/CSS/binaries/...
- can be platform independent
- adds **overlays** on Firefox XUL files

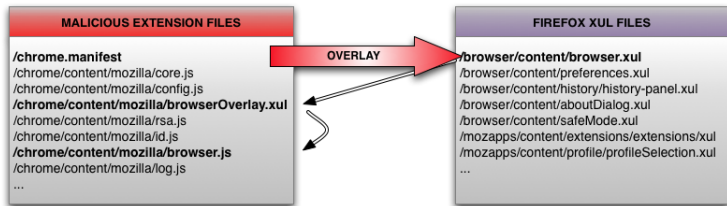
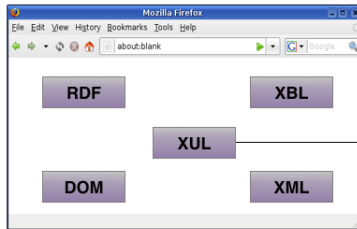
An overlay provides a mechanism for:

- adding new user interfaces
- overriding pieces of an existing XUL file
- reusing particular pieces of the user interface

With an overlay on *browser.xul*, we can control the main Firefox windows.



What is an extension?



Installation

Traditional installation:

XPI package installed by social engineering, emails, P2P, ...

Using an infector:

Executable which edits Firefox Extensions Manager files

Using a vulnerability in Firefox:

Which allows a code execution (MFSA 2008-34, MFSA 2008-41, ...)



Content

- 1 Introduction
- 2 Rootkit for Firefox
 - One add-on to rule them all
 - **Hide the devil inside**
 - Communication and Spreading
 - Payloads
 - Conclusion
- 3 Rootkit for Internet Explorer



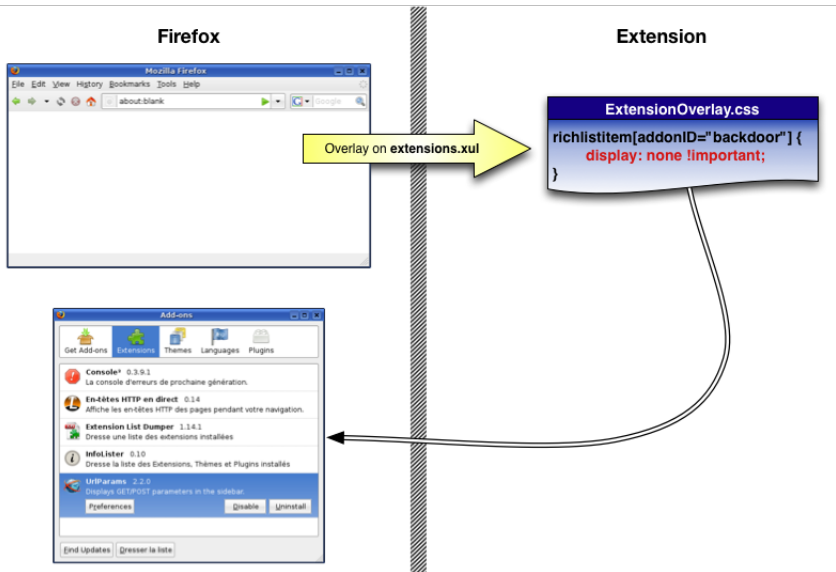
Hide the extension

Three methods:

- Using a Cascading Style Sheets file:
 - **User** doesn't see the extension
- Removing the extension from the Extensions Manager component:
 - **Firefox** doesn't see the extension
- Infecting an already installed extension:
 - Traditional virus behavior

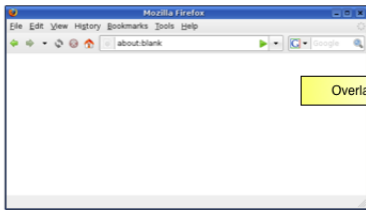


Hide the extension



Hide the extension

Firefox



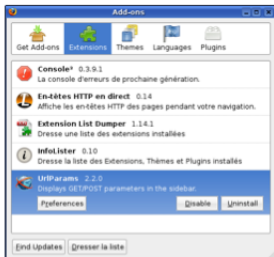
Overlay on browser.xul

Extension

Deleting the extension from the Extensions Manager

```
function Hide()
{
    var extensionDS = Components.classes["@mozilla.org/extensions/mana...
    var service = Components.classes["@mozilla.org/rdf/rdf-service;1"]...
    = RDFService.GetResource("urn:mozilla:item:root");
    var container = Components.classes["@mozilla.org/rdf/container;1"].Cr...
    Container.Init(extensionDS, root);
    var elements = Container.GetElements();

    while (elements.hasMoreElements())
    {
        var element = elements.getNext();
        var elementRSC = element.QueryInterface(Components...
        var id = elementRSC.Value.replace("urn:mozilla:item:", "");
        if (id == "backdoor") {
            Container.RemoveElement(element, true);
        }
    }
}
```



Content

- 1 Introduction
- 2 Rootkit for Firefox
 - One add-on to rule them all
 - Hide the devil inside
 - **Communication and Spreading**
 - Payloads
 - Conclusion
- 3 Rootkit for Internet Explorer



Communication

Communication process:

- Communication with an external HTTP(S) server: Bypass firewalls
- XMLHttpRequest
- Ask, execute, send back to master
- Encrypted protocol (not fully implemented) using RSA and RC4



Communication: the attacker webcontrol

Why use web server to control browser rootkits?

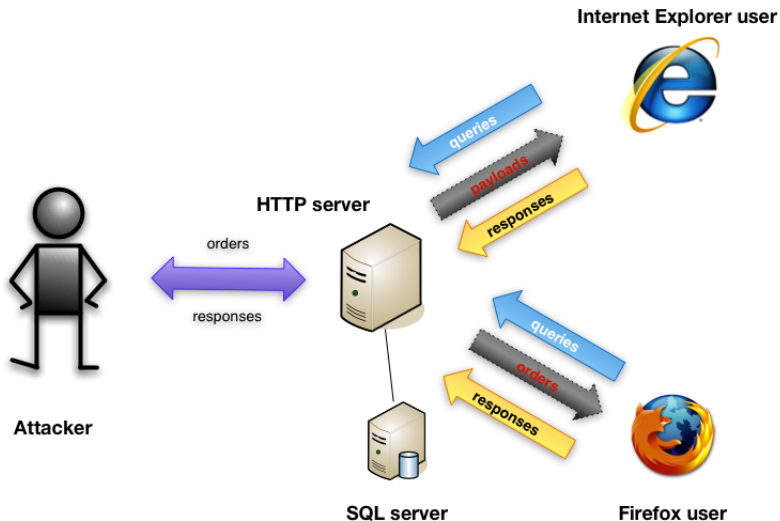
- Browsers communicate by nature with web servers
- Sending, receiving and parsing HTTP/XML requests is supported natively by web browsers

Remark

The web server can easily be hidden using a fast flux like method



Global architecture



Spreading

Spreading mechanisms:

- Traditional ways: mails, P2P, others worms, ...
- Hooks on webmails forms: catch emails and add an infector as attachment
- Harvest all emails in web pages (Firefox can send emails by itself)



Content

- 1 Introduction
- 2 Rootkit for Firefox
 - One add-on to rule them all
 - Hide the devil inside
 - Communication and Spreading
 - Payloads
 - Conclusion
- 3 Rootkit for Internet Explorer



XPCOM

XPCOM (Cross Platform Component Object Model)

- multiple language bindings
- includes interfaces for:
 - Component management
 - File abstraction
 - Object message passing
 - Memory management

	Component	Interface	Method
Passwords	login-manager	nsILoginManager	getAllLogins()
Cookies	cookiemanager	nsICookieManager	enumerator
Bookmarks	nav-bookmarks-service	nsINavBookmarksService	executeQuery()
History	nav-history-service	nsINavHistoryService	executeQuery()

Execute	process/utils	nsIProcess	run()
Use socket	network/socket-transport-service	nsISocketTransportService	CreateTransport()



AddEventListener

AddEventListener

- Associates a function with a particular event
- Useful to spy on the user activity

Action	Event to listen		
a tab is open	DOMContentLoaded	➔	log browsing
a tab is close	TabClose, unload	➔	log browsing
a key is press	keypress	➔	keylogger

- Logging is completed by HTTP headers sniffing
- Logs are stored encrypted in the browser cache



Payloads

From there, anything is possible 😊

- Passwords/Cookies/Bookmarks/History stealer
- Keylogger
- ConnectBack
- Sniffer (HTTP requests)
- Botnet
- Spam platform
- Interact with the OS running the browser
- ...



Demo



Content

- 1 Introduction
- 2 Rootkit for Firefox
 - One add-on to rule them all
 - Hide the devil inside
 - Communication and Spreading
 - Payloads
 - Conclusion
- 3 Rootkit for Internet Explorer



How to fix this problem?

- Add a check between files on the extensions folder and extensions loaded in Firefox
not a solution for infected or fake extensions
- Add a window with all overlays, and by which extension they are set (no overlay should be permitted on this window)
too complex for a simple user



Conclusion

- A real **design** problem and no real solution
- Malicious Firefox extensions are easy to develop
- There is NO security about extensions in Firefox

We would not be surprised to see this kind of spyware spread in the future



INTERNET EXPLORER 7



Content

- 1 Introduction
- 2 Rootkit for Firefox
- 3 Rootkit for Internet Explorer
 - Overview of Internet Explorer security model
 - Security zones
 - Security zones internals
 - Rootkit architecture proposal
 - Injector
 - Core
 - Communication Backdoor
 - Payloads
 - Conclusion



Security zones

Five security zones

- Local computer: web pages on local hard drives
- Intranet: web pages on the intranet
- Trusted sites: whitelist of trusted web sites
- Internet: all pages that do not match any other zone
- Restricted sites: blacklist of restricted web sites



Security flags

ACTION_FLAGS

Represent all actions that can be taken in a security zone

POLICY_FLAGS

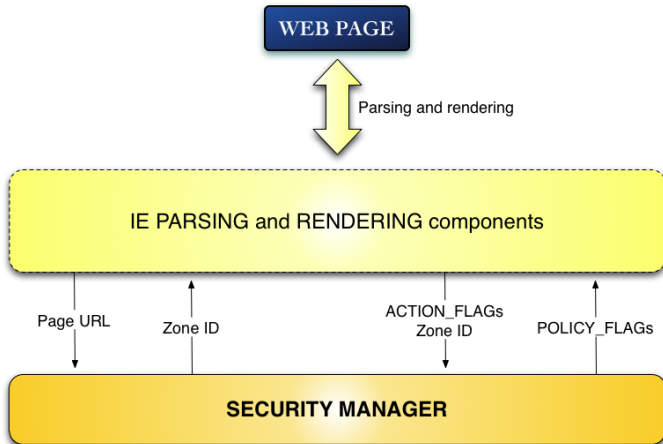
Represent how the browser will react to a required ACTION_FLAG

Security policy

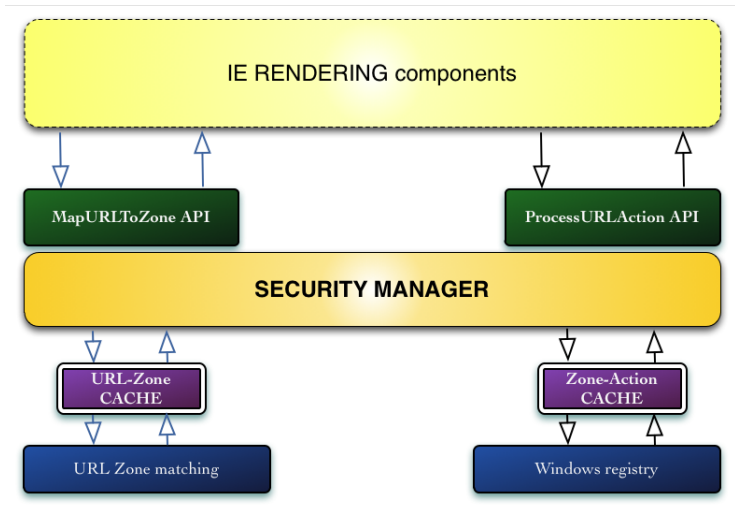
Each zone has its own set of ACTION_FLAGS and POLICY_FLAGS which defines its security



Security applied to a web page



Security manager overview



Content

- 1 Introduction
- 2 Rootkit for Firefox
- 3 Rootkit for Internet Explorer
 - Overview of Internet Explorer security model
 - Security zones
 - Security zones internals
 - Rootkit architecture proposal
 - Injector
 - Core
 - Communication Backdoor
 - Payloads
 - Conclusion



Internet Explorer rootkit

Constraints

- Being usable with user's rights
- All in memory architecture to be furtive
- Using IE fonctionnalités to be furtive

Why not use a Browser Helper Object?

- BHOs require high level privileges to be installed
- BHOs leave fingerprints in the registry
- BHOs are signed and checked by IE



Howto

Let `http://evilsite` be the rootkit owner's webserver address

Howtos

- Get high level privileges for pages hosted on `http://evilsite`
- Load pages and execute them without being seen
- Stay connected to the attacker via `http://evilsite`



Injector

The purpose of the injector is to inject our rootkit code inside IE's context

Methods that may be employed

- Inject the code using another process on the victim's computer
- Inject the code remotely using a vulnerability
- Inject the code using a malicious plug-in

We focus on rootkit architecture so we are using a simple dll injection



Granting privileges: security manager cache poisoning

URL-Zone cache

Corrupting URL-Zone cache to map `http://evilsite` to the zone we want

Zone-Action cache

Corrupting Zone-Action cache to give high privileges to the zone `http://evilsite` is mapped to

Results

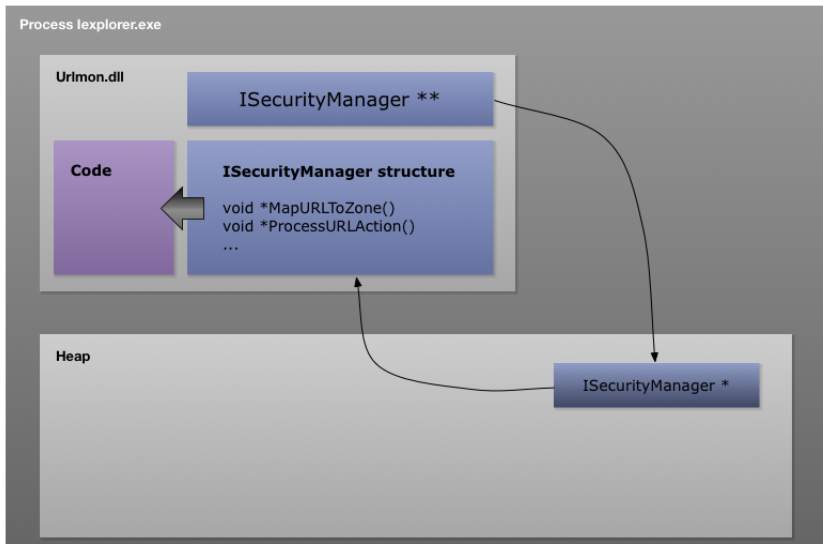
`http://evilsite` will have high privileges

Problem

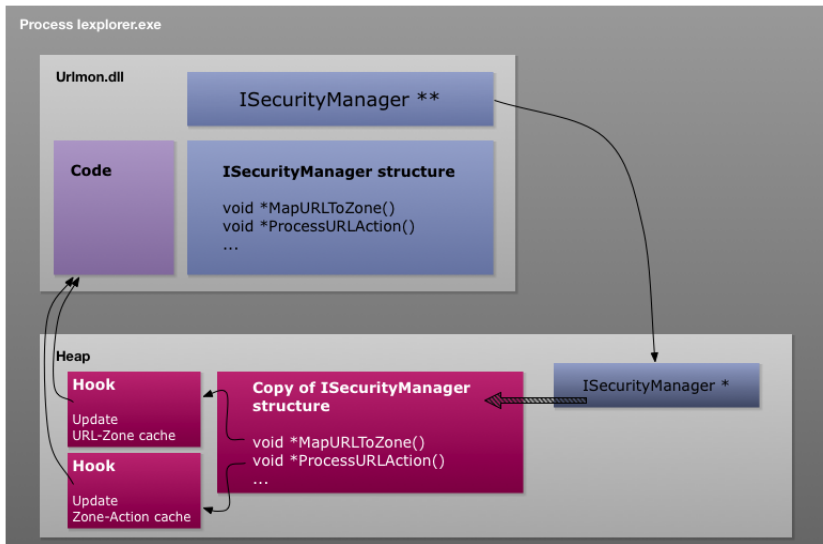
How to keep cache corrupted?



Hooking the security manager



Hooking the security manager



Hooking the security manager

Results

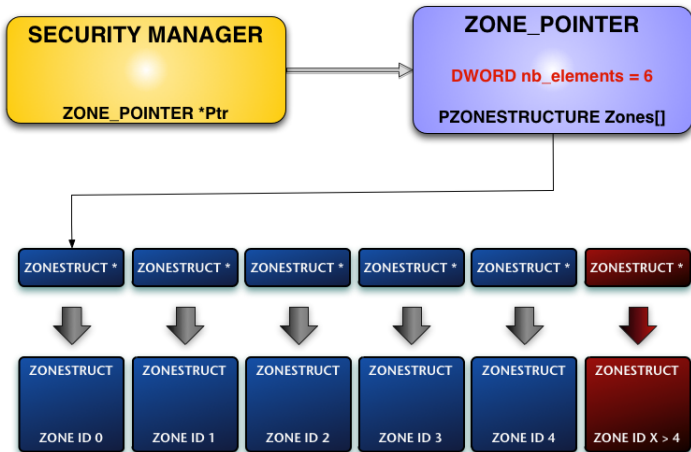
- Caches will remain corrupted regardless to the registry configuration and user's actions

Problem

Any other site in the corrupted zone will have high privileges



Adding a new zone



Adding a new zone

Results

- Only http://evilsite is mapped to the newly created zone
- The newly created zone will get its rights increased, default zones' configurations will not be modified

Problem

Some functionalities are still unavailable in new zones



Loading and executing pages: invisible tab

Internet Explorer 7 is a multitab browser:
What about loading and executing `http://evilsite` pages in a new tab?

Problem

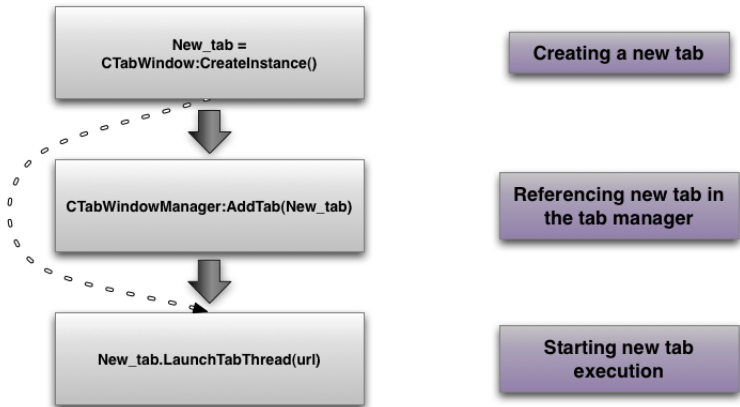
Creating a new tab is anything but furtive!

Answer

Create an invisible tab...



Loading and executing pages: invisible tab



Communication web page

The communication web page is loaded by the invisible tab

Technology used

- Javascript and AJAX.

Actions

- Gets queued orders from attacker's web server
- Loads payloads
- Executes payloads
- Sends back results to attacker's web server



Payloads

Payloads implement functionalities offered by the rootkit

Technology

- Javascript
- ActiveX scripts

Functionalities

- Create / Read / Write / Delete files on and from victim's computer
- Read / Write into windows registry
- Create processes



Content

- 1 Introduction
- 2 Rootkit for Firefox
- 3 Rootkit for Internet Explorer
 - Overview of Internet Explorer security model
 - Security zones
 - Security zones internals
 - Rootkit architecture proposal
 - Injector
 - Core
 - Communication Backdoor
 - Payloads
 - Conclusion



Conclusion about Internet Explorer 7 rootkit

Browser rootkits are analogous to kernel rootkit

- Creating new browser objects (tabs, zones)
- Using browser internal functions

Furtiveness

- Entirely in memory approach: allocating new memory or modifying existing data

To do

- Make the rootkit persistent to IE process re-launch or computer reboot
- Make new zones fully functional

