EyjafjallajöKull Framework
(aka: Exploit Kits Krawler Framework)

Seeking Exploit Kits at Large Scale Made Easy

By **Sébastien Larinier** / **@Sebdraven** & **Guillaume Arcas** / **@y0m**

sekoia
#CYBERSECURITY

botconf
First botnet fighting conference 2013

This Slide Intentionally Left (almost) Blank.

Who Are We?

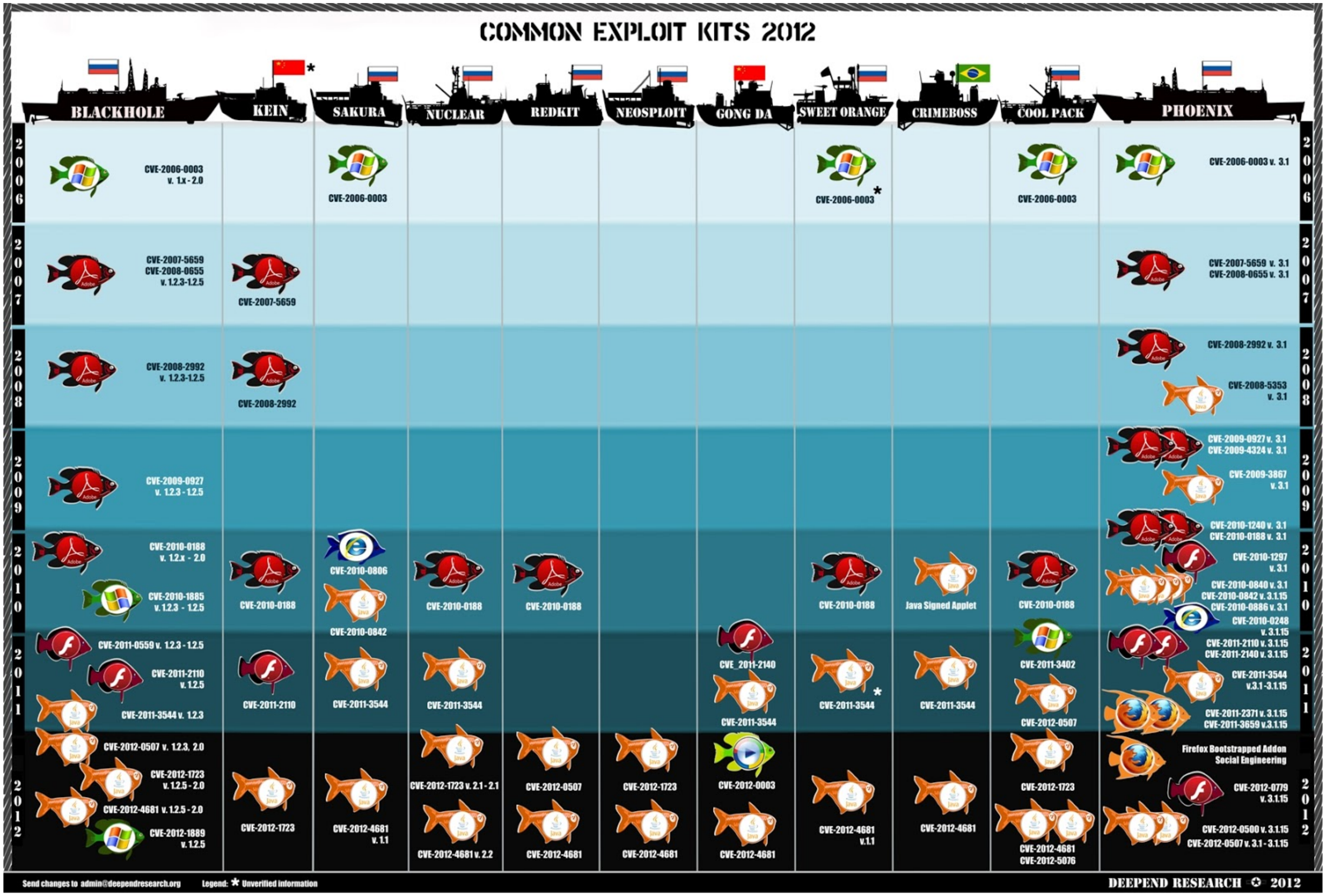
- Curious guys
- Experience in Network Analysis (we ❤ PCAP!)
- and Python coding (well... Especially Sébastien)

Sorry, We are French!

From Russia with Sploits

What is an Exploit Kit (EK, sometimes also called Exploit Pack)?

- Malicious software used to conduct "drive-by" attacks
- Targeting flaws in browsers & add-ons/plugins (most often Java, PDF, Flash)
- User just has to browse a malicious page to get infected if his/her browser is vulnerable
- Used to spread banking malware (ZeuS, etc) but also during APT attacks #BuzzWord



COMMON EXPLOIT KITS 2012

	BLACKHOLE	KEIN*	SAKURA	NUCLEAR	REDKIT	NEOSPLOIT	GONG DA	SWEET ORANGE	CRIMEBOSS	COOL PACK	PHOENIX
2006	CVE-2006-0003 v. 1.x - 2.0		CVE-2006-0003					CVE-2006-0003*		CVE-2006-0003	CVE-2006-0003 v. 3.1
2007	CVE-2007-5659 CVE-2008-0655 v. 1.2.3-1.2.5	CVE-2007-5659									CVE-2007-5659 v. 3.1 CVE-2008-0655 v. 3.1
2008	CVE-2008-2992 v. 1.2.3-1.2.5	CVE-2008-2992									CVE-2008-2992 v. 3.1 CVE-2008-5353 v. 3.1
2009	CVE-2009-0927 v. 1.2.3 - 1.2.5										CVE-2009-0927 v. 3.1 CVE-2009-4324 v. 3.1 CVE-2009-3867 v. 3.1
2010	CVE-2010-0188 v. 1.2.x - 2.0 CVE-2010-1885 v. 1.2.3 - 1.2.5	CVE-2010-0188	CVE-2010-0806 CVE-2010-0842	CVE-2010-0188	CVE-2010-0188			CVE-2010-0188	Java Signed Applet	CVE-2010-0188	CVE-2010-1240 v. 3.1 CVE-2010-0188 v. 3.1 CVE-2010-1297 v. 3.1 CVE-2010-0840 v. 3.1 CVE-2010-0842 v. 3.1.15 CVE-2010-0886 v. 3.1 CVE-2010-0248 v. 3.1.15
2011	CVE-2011-0559 v. 1.2.3 - 1.2.5 CVE-2011-2110 v. 1.2.5 CVE-2011-3544 v. 1.2.3	CVE-2011-2110	CVE-2011-3544	CVE-2011-3544			CVE_2011-2140 CVE-2011-3544	CVE-2011-3544*	CVE-2011-3544	CVE-2011-3402 CVE-2012-0507	CVE-2011-2110 v. 3.1.15 CVE-2011-2140 v. 3.1.15 CVE-2011-3544 v.3.1 -3.1.15 CVE-2011-2371 v. 3.1.15 CVE-2011-3659 v.3.1.15
2012	CVE-2012-0507 v. 1.2.3, 2.0 CVE-2012-1723 v. 1.2.5 - 2.0 CVE-2012-4681 v. 1.2.5 - 2.0 CVE-2012-1889 v. 1.2.5	CVE-2012-1723	CVE-2012-4681 v. 1.1	CVE-2012-1723 v. 2.1 - 2.1 CVE-2012-4681 v. 2.2	CVE-2012-0507 CVE-2012-4681	CVE-2012-1723 CVE-2012-4681	CVE-2012-0003 CVE-2012-4681	CVE-2012-4681 v.1.1	CVE-2012-4681	CVE-2012-1723 CVE-2012-4681 CVE-2012-5076	Firefox Bootstrapped Addon Social Engineering CVE-2012-0779 v. 3.1.15 CVE-2012-0500 v. 3.1.15 CVE-2012-0507 v. 3.1 - 3.1.15

Send changes to admin@deependresearch.org   Legend: * Unverified information

DEEPEND RESEARCH © 2012

Source: http://www.deependresearch.org/

BlackHole Exploit Kit

- Born on 2010
- Coded by "Paunch" and ""HodLuM"
- One of the most popular EK ever
- PHP + HTML + JavaScript
- Exploits for Java + PDF + Flash + IE + MS Windows
- Exploits updated on a daily basis
- Advanced Obfuscation Techniques (#BuzzWord) for JS & PDF
- URLs spread by spam campaigns
- SaaS business model ($1500 / year)

Bad Times for Bad Guys

- Phoenix Exploit Kit author arrested in Russia in April, 2013
- One of the BHEK authors arrested in Russia in October, 2013

Why Studying Exploit Kits?

- Look for similarities: do some EKs "share" same exploits? If yes, which ones?
- Understand URLs diffusion methods, especially when URLs are spread in webpages
- Understand targeting system: which countries, which browsers are targeted, which malware are sent?
- Understand Obfuscation methods
- Mapping EK targets & payloads
- Identifying EK authors (... just joking!)

How to Find EK - The Lazy Way

1. Browse http://www.malwaredomainlist.com/update.php
2. Pick a URL & pray for it to be still active
3. Run a VM embedding a supposedly vulnerable browser
4. Open the URL from the VM
5. Cross fingers & see if the VM gets infected.

Well, it looks easy!

But failure can occur at each of these steps...

- URL can already be offline
- Triggers only if request is coming from a speficic page (Referer)
- Or with "valid" Cookies
- Only triggers once: the next request from the same IP will be discarded
- Only triggers if User-Agent matches with available exploits
- Only triggers if IP belongs to a specific geographic area
- Use of Evasion & Obfuscation techniques
- Use of Anti-robot & Anti-spider techniques
- Check that a human is browsing the malicious page.

So, it looks like wget or curl won't fit...

Automating EK Browsing?

Challenge accepted!

What Do We Need?

- Finding malicious webpages
- Browsing the found webpages with vulnerable browsers
- Avoiding failure (see previous slide)
- Running exploits
- A (hopefully) good coder.

Finding Malicious URLs

1. Spam Campaign
   - Using SpamBoxes
   - Extracting good candidates URLs
   - Feed a spider
2. Malicious URLs
   - Spamvertizing
   - Search Engines & keywords
   - Twitter Trends
   - Facebook Messages
3. Online submission

Browsing Malicious URLs

- What if some websites requires authentication?
- How to preserve HTTP Referer & Cookies?
- How to know what specific browsers are vulnerable?
- Geolocation

Running Exploits & Payloads

- OK, my browser is vulnerable but what kind of malware is run?
- In some cases, a same malicious page distributes different payload: trojan horse or ransomware

Why EK Krawler Framework?

- It's easier to pronounce that EyjafjallajöKull Framework.
- Is it a spider? No, it's Selenium driven browsers fed with different sources.
- Is it a sandbox? No, it is a collection of VMs from various types
- Is it a proxy? Kind-of, but collecting all objects (files, HTTP headers, etc) with SSL MiTM capabilities
- ***It's Exploit Kit Krawler!***

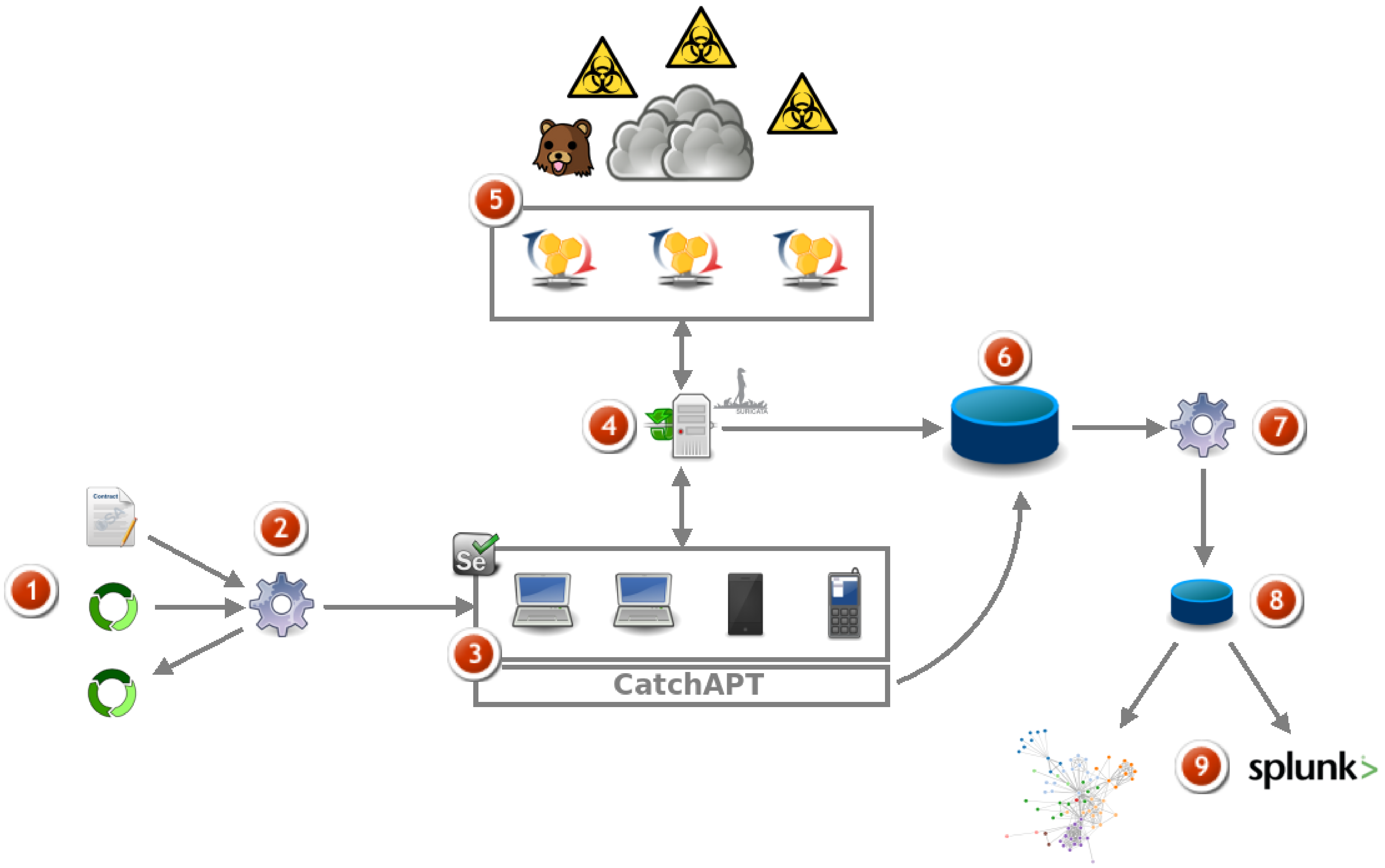
It's Spiderman, Batman & Superman working in team!
(with Robin preparing coffee)

How Do We Do That?

- Python
- Selenium
- Virtual Machines. Currently VirtualBox.
- Python again
- HoneyProxy (well, Python, once more...)
- Reddis & MongoDB for data storage

圖勝萬言

"One picture worth thousands words." (Chinese proverb)



5
6
4
SURICATA
7
Contract
2
1
Se
8
3
CatchAPT
9
splunk>

1. URLs: from files, grabbed on twitter/facebook/google, submitted, from logfiles
2. Dispatching engine: sends URls to appropriate VMs (Selenium)
3. Pool of VMs from various types (note: the Vms may be dispatched on different location)
4. Pcap Factory: capture network traffic from/to the VMs, processes it with Suricata
5. Honeyproxy instances, geographically dispatched (exit nodes)
6. Big database: store all collected artifacts (files, http requests, exploits, etc)
7. Posst-processing (data reduction, correlation)
8. Smaller database

9. Visualisation interfaces

Lot of stuff still "under coding"

Demo

Todo List

- Multi-hypervisor support
- Front-end Web "à la urlQuery"
- Integration of VADtools
- JS Deobfuscation
- Bubbling output

- Host servers
- Host exit nodes (socks proxy)

Thank You!