Monitoring web sites for malware injection with WebDetector

A pet project from Paolo Di Prodi

OWASP

The Open Web Application Security Project

About Me

OWASP
The Open Web Application Security Project

- Presenting today:
  - Dr. Paolo Di Prodi a.k.a “epokh”
- Previous work on:
  - Started with DOS,Win3.1,Win95,98 now Win8!
  - Binary reversing on crack and warez
  - First Java ME bytecode patcher
- Research background:
  - Machine learning and AI in social robotics

Microsoft

About Me

OWASP
The Open Web Application Security Project

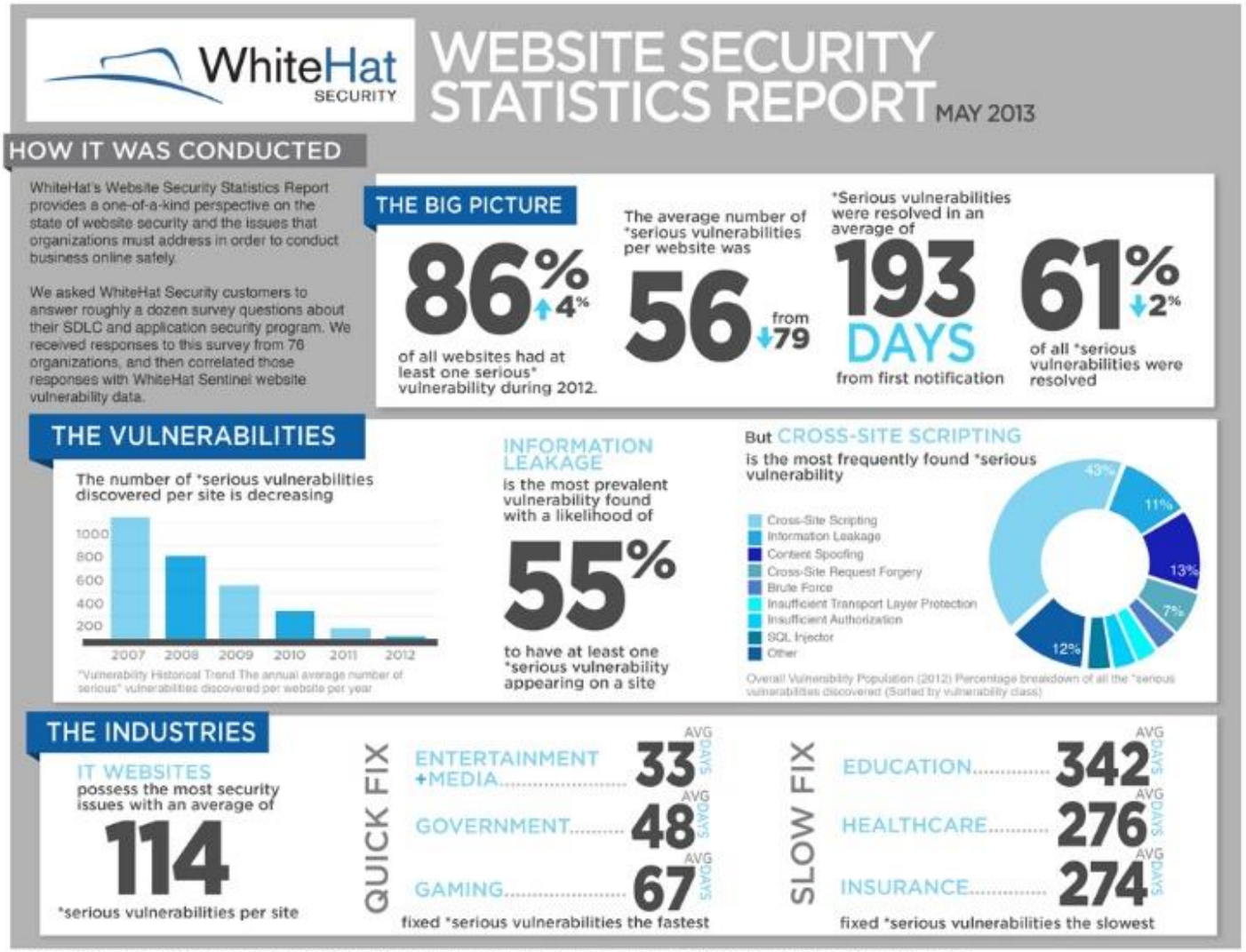
- Working now for Microsoft on:
  - Machine Learning applied to Intrusion detection
  - Big data and security
  - Build POC systems with any technology available from Microsoft and outside

Microsoft

About Me

OWASP
The Open Web Application Security Project

- Problem:
  - How many websites are compromised?
  - What are the most common attack vectors?
  - How can we monitor a website?
- Solution:
  - Requires a web tracking system
  - Requires JS sandboxing
  - Requires heuristics



Source: White Hat Report 2013

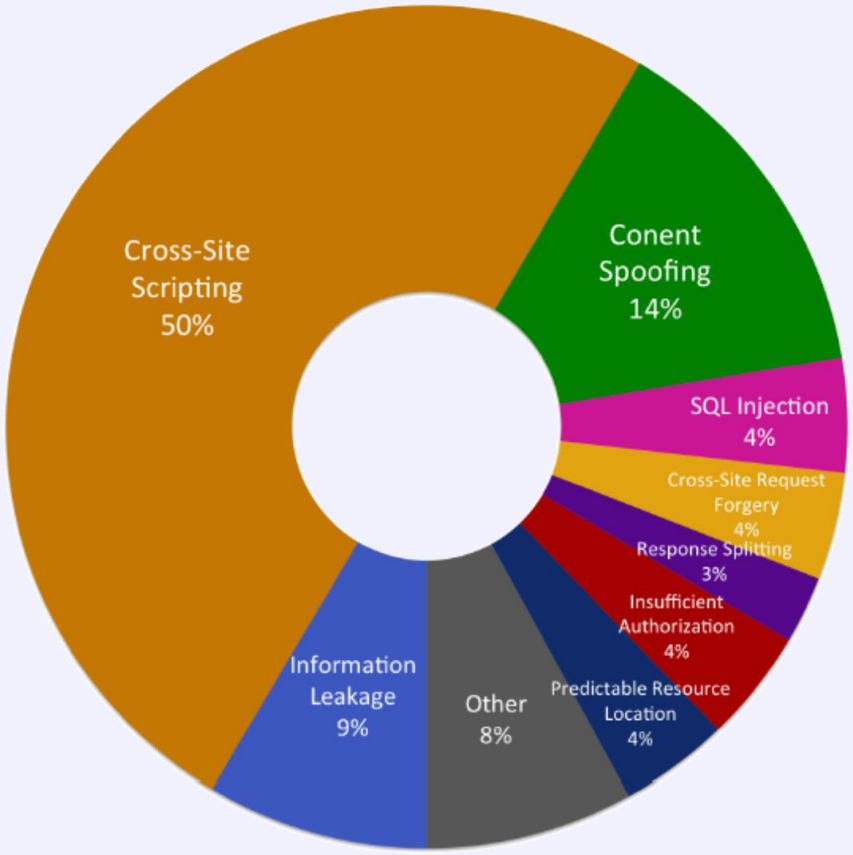


Figure 4. Overall Vulnerability Population (2011)
Percentage breakdown of all the serious* vulnerabilities discovered
(Sorted by vulnerability class)

Source: Black Hat Report 2011

XSS types:

– Stored

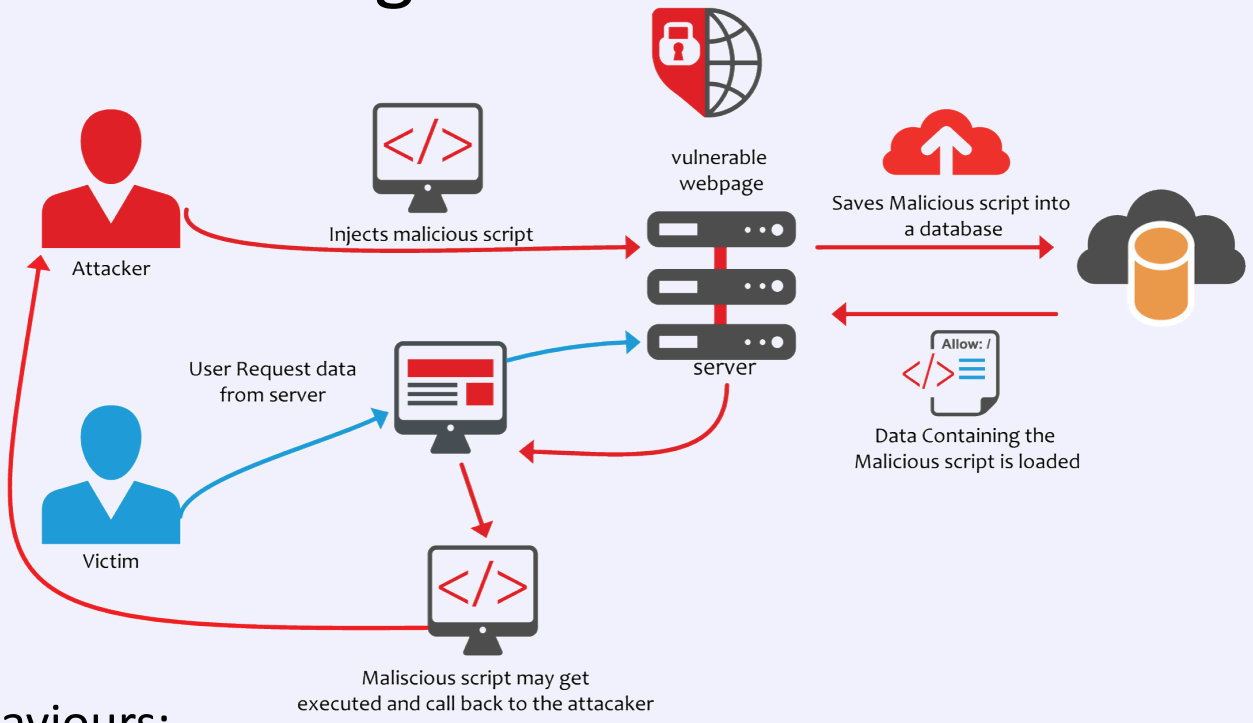
– Reflected

– Dom based

XSS vectors:

- JS
- PDF
- ActiveX
- Iframe
- Shockwave

- Example of storage based XSS



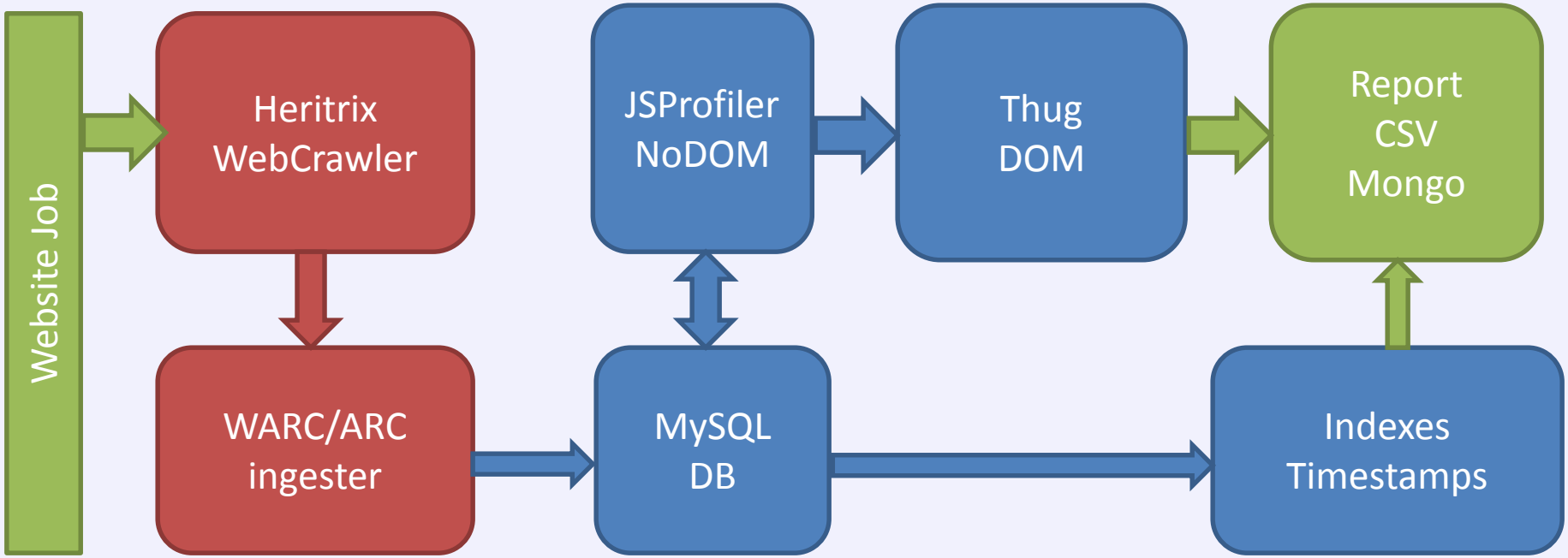
- Basic behaviours:
  - URL injection to malware dropper
  - Form injection for phishing
  - Page redirection
  - Cookie stealing

Source: Acunetix

- We need to monitor our websites:
  - We want to know:
    - What was added/removed/deleted?
    - When did that happen?
    - Who did it?
    - Is it malicious or just our webadmin playing tricks?
  - Solution:
    - Sounds like a GIT job to me
    - With some WGET scripting
  - Problems:
    - WGET not doing a good crawling job especially for dynamic pages

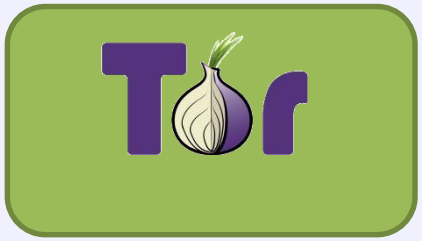

OWASP
The Open Web Application Security Project

Webtracker


Website Job
Heritrix
WebCrawler
WARC/ARC
ingester
JSProfiler
NoDOM
MySQL
DB
Thug
DOM
Report
CSV
Mongo
Indexes
Timestamps

OWASP
The Open Web Application Security Project

Webtracker



- Provision the VM -> 30 minutes
- Install packages
- Java, Python, Perl etc
- Google V8 JS patched
- Mozzilla JS patches
- Setup DB
- Setup Heritrix
- Ready to go

OWASP
The Open Web Application Security Project

- Heritrix
  - Latest Version January 2014
  - Java Based Web Crawler
  - Run on multiple instances
- Python Thug:
  - Latest Version March 2014
  - HoneyPot client sandbox for JS execution
  - Currently detects: 161 Exploits
  - Including: Adobe PDF, Shockwave, Java Web

OWASP
The Open Web Application Security Project

Patched JS monkey

- Logs the following features for each JS script
  - Document.write
  - string_instance: var foo=“hello world”
  - Element instance: var btn=document.createElement
  - Object instance: var foo={text:”hello world”}
  - Eval: var foo=eval(“x * y “)
  - Location: hash,host,hostname,href,origin,port,etc:
  - Escape and unescape
  - Encode: encodeURI(),decodeURI()
  - Decode: dencodeURIComponent()

OWASP
The Open Web Application Security Project

Useful to de-obfuscate

Example:

```
eval(function(p,a,c,k,e,d){e=function(c){return
c.toString(36)};if(!''.replace(/^/,String)){while(c--
){d[c.toString(a)]=k[c]||c.toString(a)}k=[function(e){r
eturn d[e]}];e=function(){return'\\w+'};c=1};while(c-
-){if(k[c]){p=p.replace(new
RegExp('\\b'+e(c)+'\\b','g'),k[c])}}return
p}('5("1.4(\'<0 7=\\"3\\"
2=\\"6://d.8.c/b/?a=\'+1.9+\'\\">></0>\');");',14,14,'s
cript|document|src|Javascript|write|eval|http|lan
guage|robomotic|referrer|ref|style|com|www'.spli
t('|'),0,{}))
```

OWASP
The Open Web Application Security Project

Useful to de-obfuscate

Original was:

```
eval("document.write('<script language=\"Javascript\"
src=\"http://www.robomotic.com/style/?ref='+document.referrer+'\"></
script>');");
```

Profile generated (on the obfuscated one):

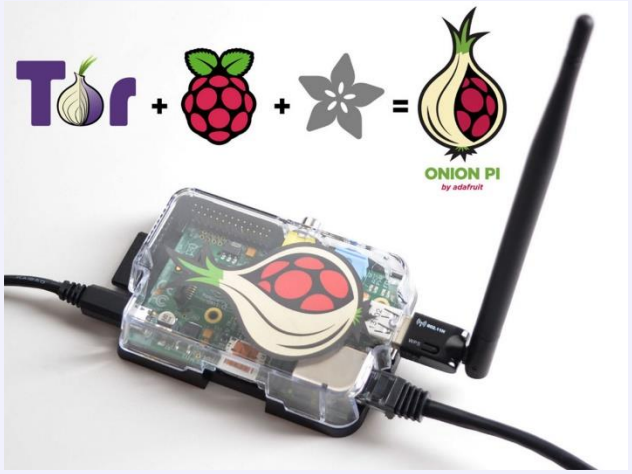
- String_instance: 4
- Document_Write: 1
- Element_instance, Object_instance,decode,location,escape,decode: 0
- Eval: 2

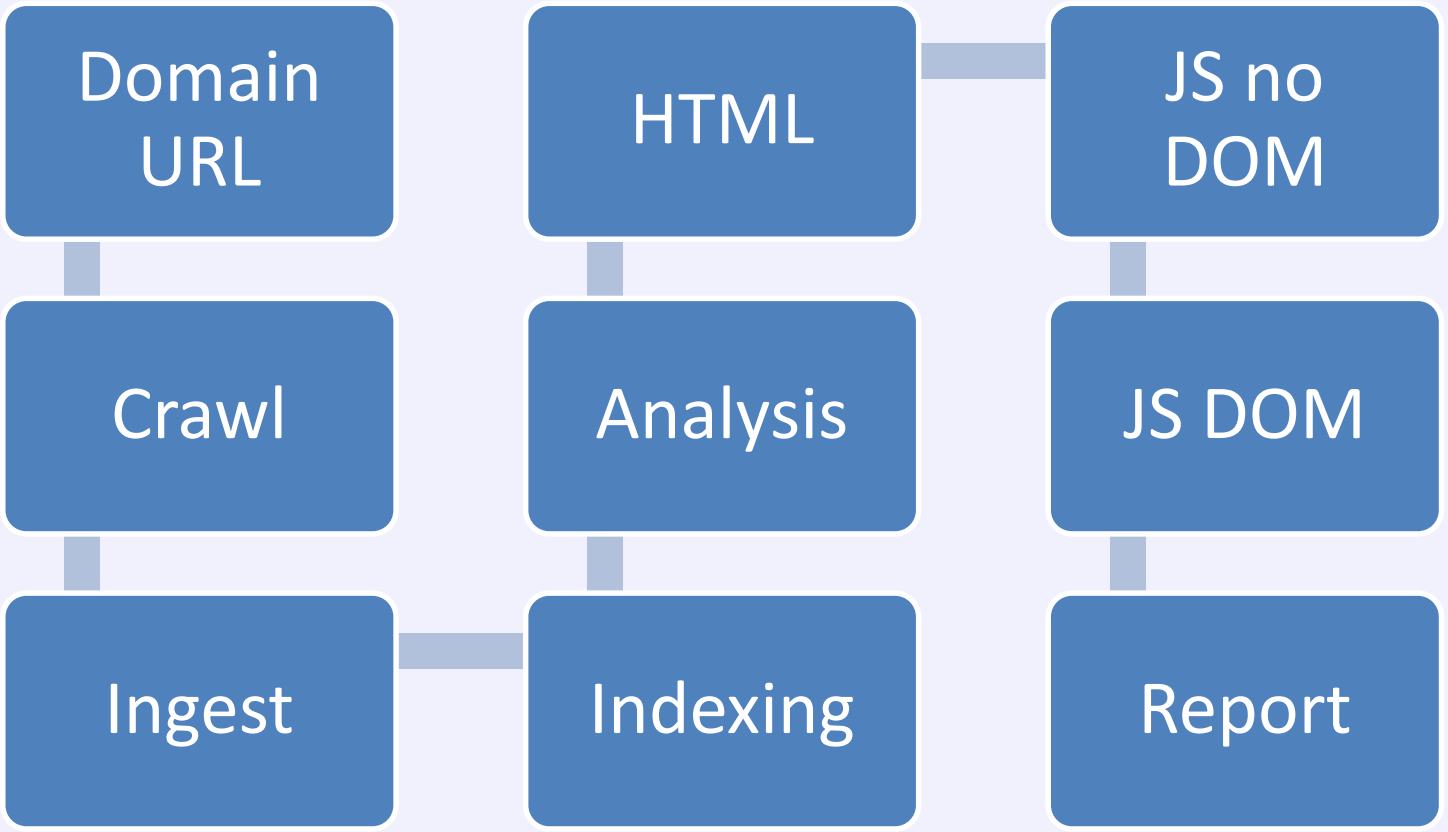
OWASP
The Open Web Application Security Project

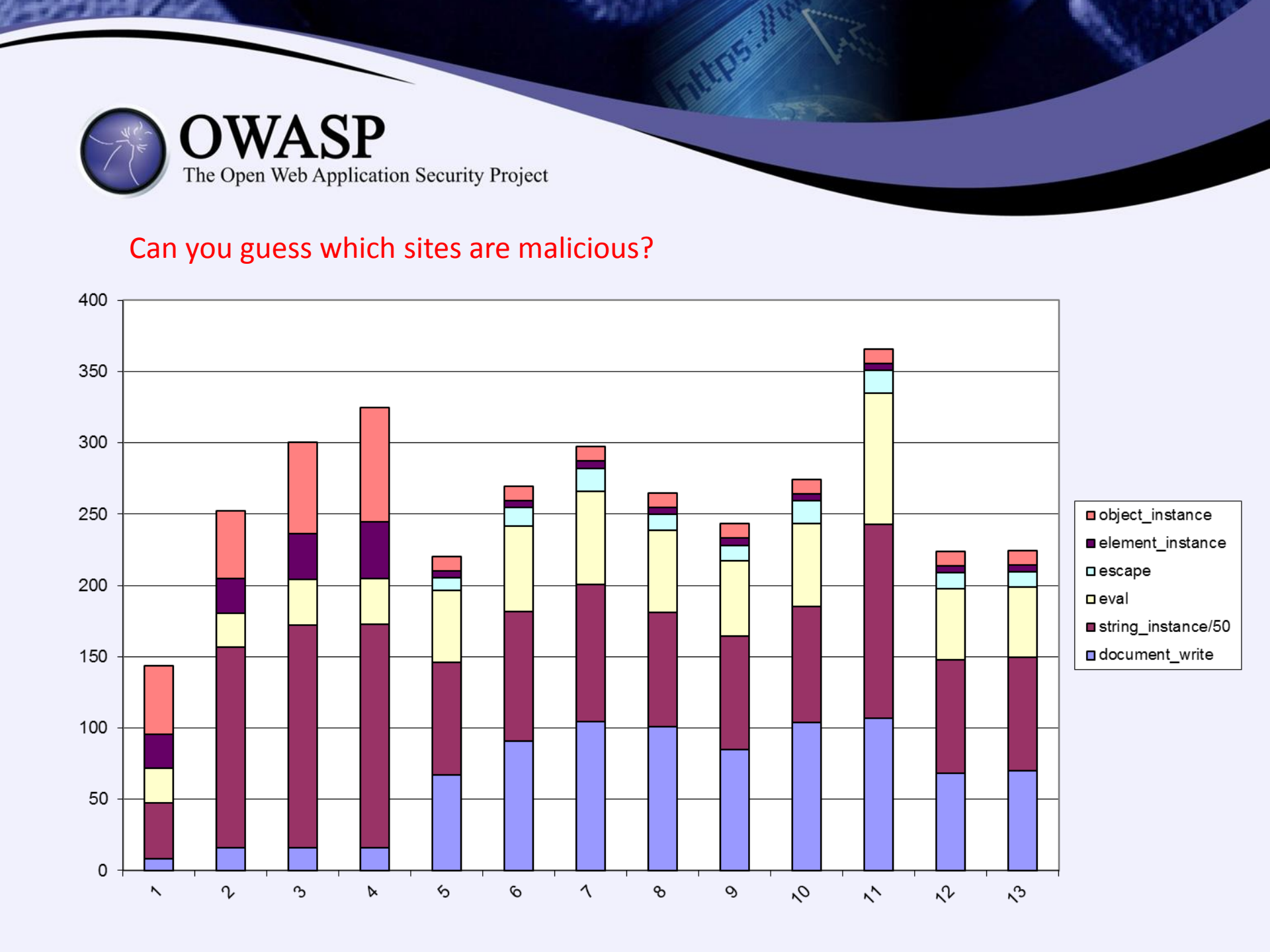
Python thug

- Logs the following features for each JS script
  - Document.write
  - string_instance: var foo=“hello world”
  - Element instance: var btn=document.createElement
  - Object instance: var foo={text:”hello world”}
  - Eval: var foo=eval(“x * y “)
  - Location: hash,host,hostname,href,origin,port,etc:
  - Escape and unescape
  - Encode: encodeURI(),decodeURI()
  - Decode: dencodeURIComponent()

- WebDetector
  - Version 3:
    - Django app with RESTFUL api to Hedrix
    - Celery for distributed task allocation
    - Integration with Zozzle
- Anonimization:
  - Raspberry PI with Onion PI
  - Basically TOR

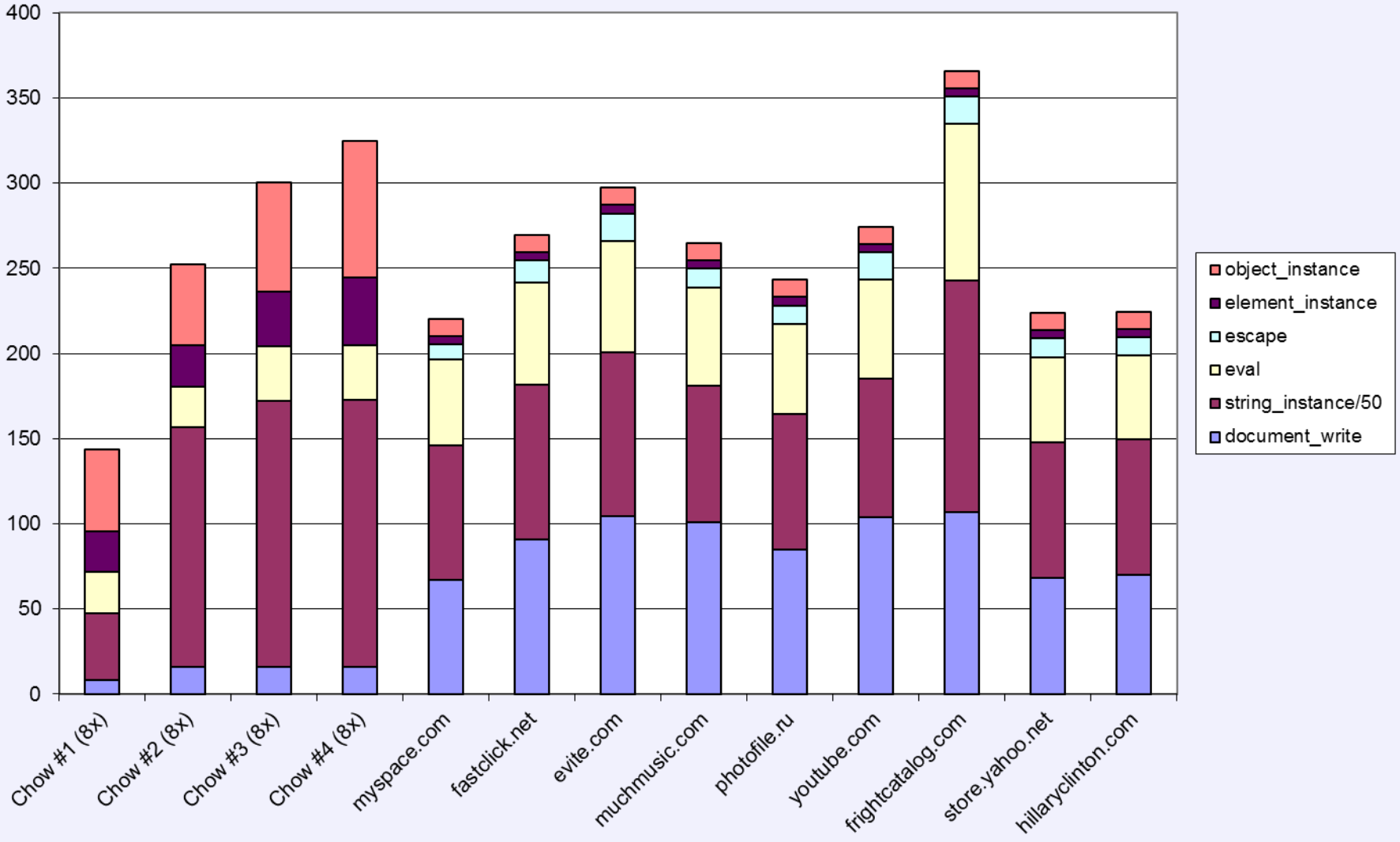



Domain URL
Crawl
Ingest
Indexing
Analysis
HTML
JS no DOM
JS DOM
Report



Can you guess which sites are malicious?

The first four were fetched by compromised malware sites



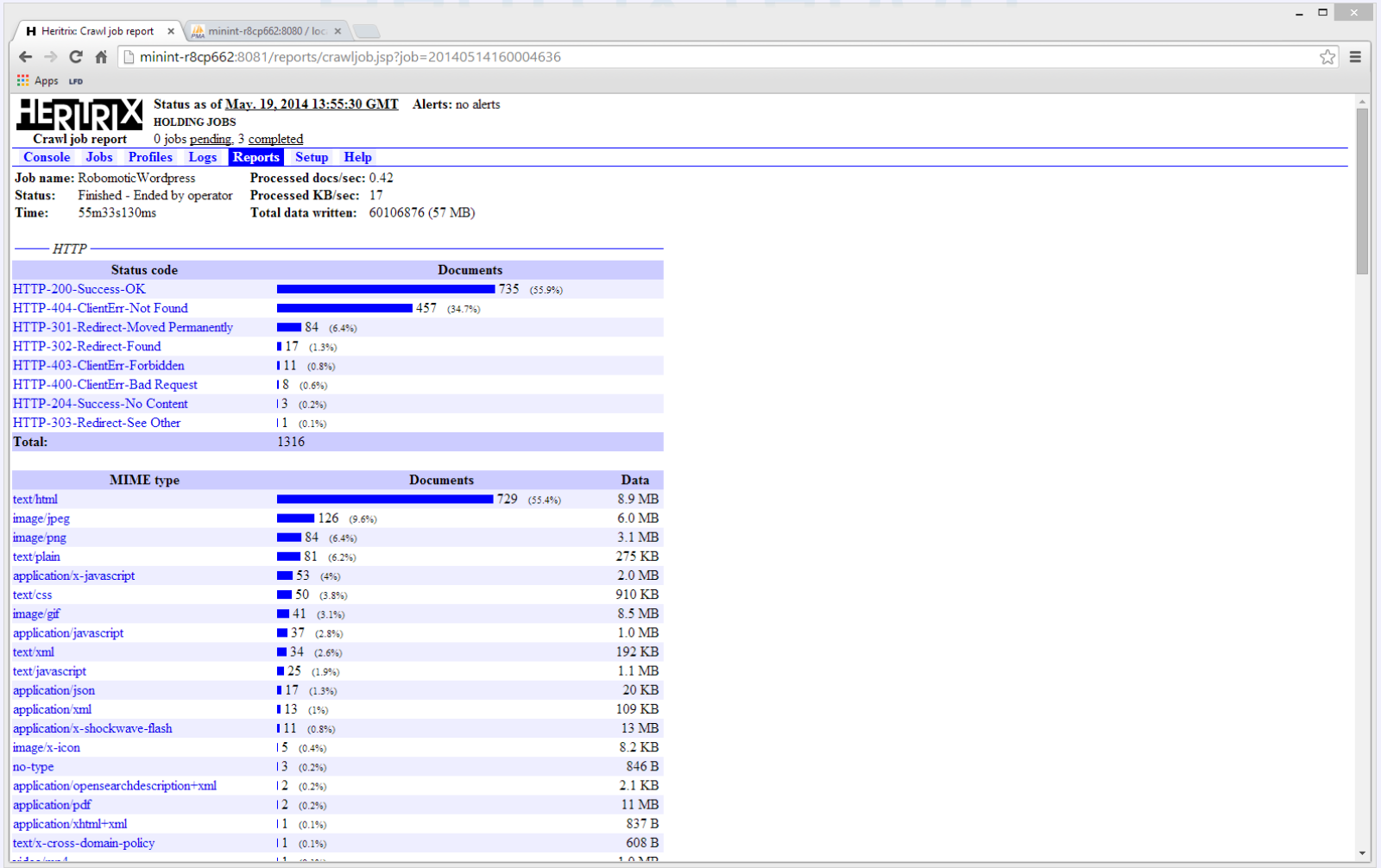
Classification via Naïve Bayes or Decision Tree gives 100% accuracy.
Small dataset: 85 GOOD, 64 BAD

Mean(Feature)	GOOD	BAD
Document_write	90	15
String_instance	4626	135
Element_instance	5	30
Object_instance	10	60
Eval	61	27
Escape	12	0

A larger dataset is required to run a statistically significant classification rule, but is a good start

OWASP
The Open Web Application Security Project

Heritrix report



Heritrix: Crawl job report | minint-r8cp662:8080 / loc

minint-r8cp662:8081/reports/crawljob.jsp?job=20140514160004636

Apps LFD

HERITRIX
Crawl job report

Status as of May. 19, 2014 13:55:30 GMT **Alerts:** no alerts
HOLDING JOBS
0 jobs pending. 3 completed

Console | Jobs | Profiles | Logs | Reports | Setup | Help

Job name: RobomoticWordpress
Status: Finished - Ended by operator
Time: 55m33s130ms
Processed docs/sec: 0.42
Processed KB/sec: 17
Total data written: 60106876 (57 MB)

HTTP

Status code	Documents
HTTP-200-Success-OK	735 (55.9%)
HTTP-404-ClientErr-Not Found	457 (34.7%)
HTTP-301-Redirect-Moved Permanently	84 (6.4%)
HTTP-302-Redirect-Found	17 (1.3%)
HTTP-403-ClientErr-Forbidden	11 (0.8%)
HTTP-400-ClientErr-Bad Request	8 (0.6%)
HTTP-204-Success-No Content	3 (0.2%)
HTTP-303-Redirect-See Other	1 (0.1%)
Total:	1316

MIME type	Documents	Data
text/html	729 (55.4%)	8.9 MB
image/jpeg	126 (9.6%)	6.0 MB
image/png	84 (6.4%)	3.1 MB
text/plain	81 (6.2%)	275 KB
application/x-javascript	53 (4%)	2.0 MB
text/css	50 (3.8%)	910 KB
image/gif	41 (3.1%)	8.5 MB
application/javascript	37 (2.8%)	1.0 MB
text/xml	34 (2.6%)	192 KB
text/javascript	25 (1.9%)	1.1 MB
application/json	17 (1.3%)	20 KB
application/xml	13 (1%)	109 KB
application/x-shockwave-flash	11 (0.8%)	13 MB
image/x-icon	5 (0.4%)	8.2 KB
no-type	3 (0.2%)	846 B
application/opensearchdescription+xml	2 (0.2%)	2.1 KB
application/pdf	2 (0.2%)	11 MB
application/xhtml+xml	1 (0.1%)	837 B
text/x-cross-domain-policy	1 (0.1%)	608 B

OWASP
The Open Web Application Security Project

- Js Profile report

domain	samples	document_write	string_instance	element_instance	object_instance	eval	location	escape	encode	decode	class
moderndevice.com	3	86	686	11	9	42	0	10	0	0	GOOD
twimg.com	3	92	1523	9	8	38	0	7	1	4	GOOD
neurdon.com	2	74	382.5	11	10	51	0	8	0	0	GOOD
norduino.com	2	89	759	16	11	42	0	9	0	0	GOOD
robomotic.com	2	92	382	11	11	40	0	9	0	0	GOOD
archive.org	1	74	2391	8	4	38	0	0	3	9	GOOD
xxxxxxx.xx	2	32	2417	28	67	18	2	4	3	10	BAD
xxxxxxx.xxx.xx	4	38	3092	32	74	19	4	3	3	10	BAD
wordpress.com	1	96	3207	4	24	32	0	1	5	13	GOOD

OWASP
The Open Web Application Security Project

- Thug report JSON example:


```
"exploits": [
    {
        "url": "about:blank",
        "cve": "CVE-2007-4391",
        "data": null,
        "description": "Server Console Overflow",
        "module": "Yahoo! Messenger 8.x Ywcvwr ActiveX"
    }
],
"behavior": [
    {
        "timestamp": "2014-05-27 15:50:02.075500",
        "cve": "CVE-2007-4391",
        "description": "[Yahoo! Messenger 8.x Ywcvwr ActiveX] Server Console Overflow",
        "method": "Dynamic Analysis"
    }
],
"url": “xxxxxxx.fr",
"timestamp": "2014-05-27 15:50:01.878054",
"connections": [],
"locations": []
```

OWASP
The Open Web Application Security Project

Suspicious URI

Job Name	URI	Timestmap	Type	Size	Offs	File	Resp	Parent
Robomotic	http://62.76.43.78/p2p/PP_detalis_726716942049.pdf...	14/05/2014 16:14	text/html; charset='	13428676	2124	CM.1.arco	403	http://www.robomotic.com/android/ text/html

Outgoing Links

Job Name	URI	Timestmap	Type	Size	Offset	File	Res	Parent
Robomotic	http://beatdiabetes.us/	14/05/2014 16:14	text/html; charset="utf-8"	8555635	6852	CM.2.arco	200	http://www.robomotic.com/android/ text/html
Robomotic	http://beatdiabetes.us/	14/04/2014 14:37	text/html; charset="utf-8"	11544973	6852	CM.1.arco	200	http://www.robomotic.com/android/ text/html

Form redirection

Job Name	URI	Timestmap	Type	Size	Offset	File	Res	Parent
xxxxxx	http://xxxxxx	14/05/2014 18:12	text/html; charset="utf-8"	4356	234	CM.1.arco	200	http://xxxxxxxx.com/customers/submit.php

www.robomotic.com/android/

exact geographical GPS location.

BetaLocator is more advanced than an Airbag, because you can setup the threshold for the impact detection.

The detector is based on a simple but effective AI approach which uses artificial neurons (yes like the one you have in your brain) to reliably detect the impact.The basic version in the Android market does not contain the GPS location feature, which is included in the commercial one that costs only 4 GBP.

The free version is available on the android webpage.
The full version is available on the android webpage.

Follow us on Twitter

Written by bonzo BetaLocator |

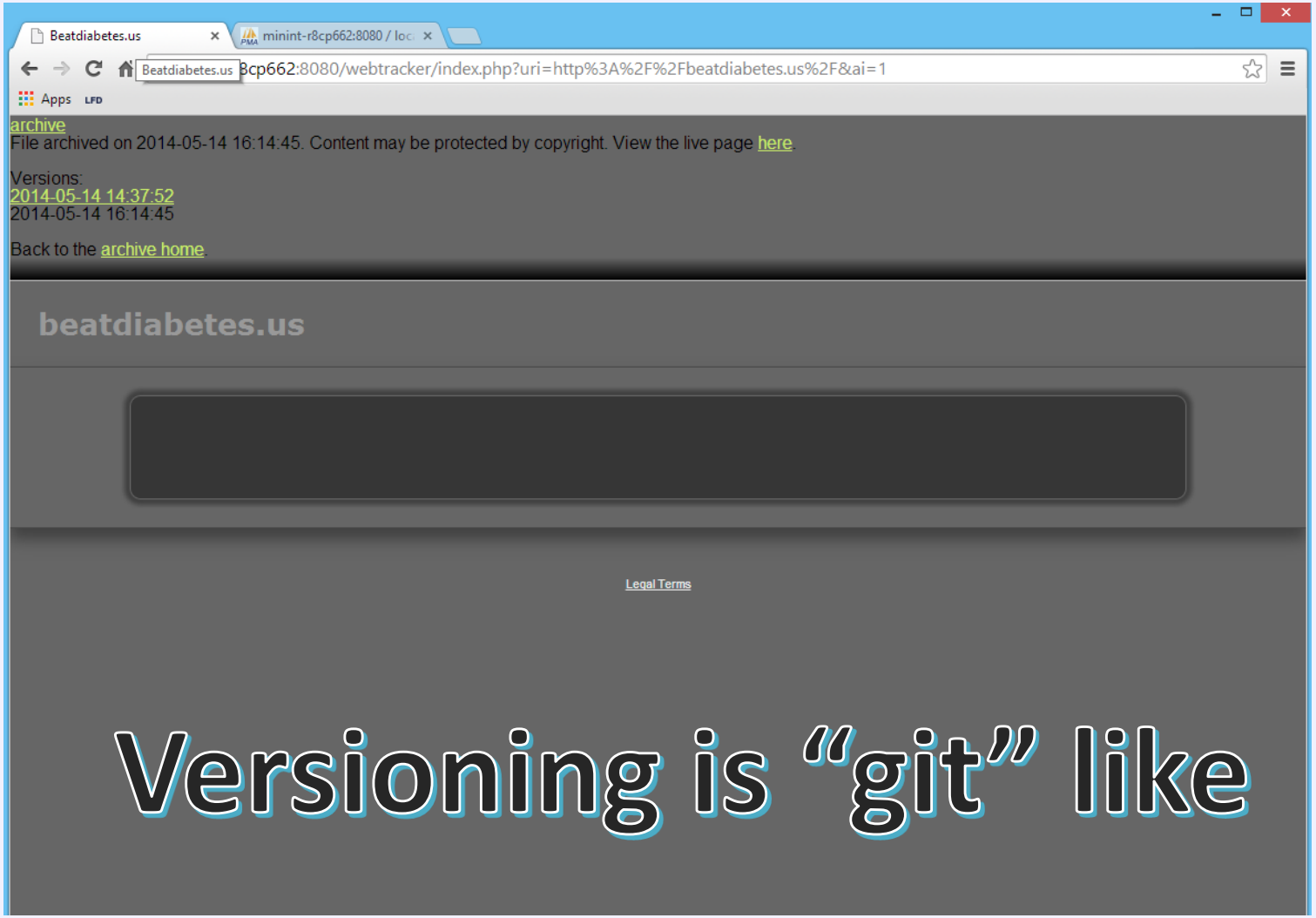
Tags

Close block

Blogroll

Close block

- Documentation
- Download App
- Plugins
- Suggest Ideas
- Support Forum
- Themes
- WordPress Blog
- WordPress Planet


Beatdiabetes.us
minint-r8cp662:8080 / loc
Beatdiabetes.us
8cp662:8080/webtracker/index.php?uri=http%3A%2F%2Fbeatdiabetes.us%2F&ai=1
Apps
LFD
archive
File archived on 2014-05-14 16:14:45. Content may be protected by copyright. View the live page here.
Versions:
2014-05-14 14:37:52
2014-05-14 16:14:45
Back to the archive home
beatdiabetes.us
Legal Terms
Versioning is “git” like

OWASP
The Open Web Application Security Project

Risk indication summary

domain	samples	outside_form	outside_ip	outside_js	inside_js	embedded_js	exploit	risk
youtube.com	151	0	0	0	161	511	0	LOW
robomotic.com	150	0	0	430	1418	630	0	LOW
xxxxxxxx.fr	104	104	0	520	624	416	2	HIGH
wordpress.org	25	0	0	4	70	136	0	LOW
lakiscamp.gr	23	0	0	13	352	98	1	HIGH
facebook.com	22	0	0	0	0	132	0	LOW
xxxxxxxx.com	15	1	0	1	2	3	1	HIGH
vimeo.com	15	0	0	32	32	32	0	LOW
gmpg.org	14	0	0	0	0	0	0	LOW
xxxxxxx.fr	5	1	0	25	25	10	1	HIGH
adobe.com	4	0	0	0	60	24	0	LOW
apple.com	4	0	0	0	44	20	0	LOW
browsehappy.com	4	0	0	6	8	6	0	LOW
neurdon.com	4	0	0	0	8	8	0	LOW
ytimg.com	4	0	0	0	0	0	0	LOW
amarino-toolkit.net	2	0	0	10	10	8	0	LOW
beatdiabetes.us	2	0	0	4	8	8	0	LOW
example.com	2	0	0	0	0	0	0	LOW
gstatic.com	2	0	0	0	0	0	0	LOW
twitter.com	2	0	0	0	0	6	0	LOW
w3.org	2	0	0	0	0	0	0	LOW
wp.com	2	0	0	0	0	10	0	LOW
xxxx.co.uk	1	0	1	23	8	8	0	MEDIUM

OWASP
The Open Web Application Security Project

- Things that I learned:
  - Wget vs Heritrix:
    - Fails to crawl highly dynamic pages
    - Efficient on simple sites
    - Plenty of rules and plugins to use
    - Heritrix requires Java environment
  - Tor: uses only socks, requires a proxy on host
  - Tor: be careful with DNS leaking

OWASP
The Open Web Application Security Project

- Detecting more XSS passive:
  - Harder than active XSS scanning
    - Semantically how do you know that a web page is behaving as originally designed?
    - Efficient on simple sites
    - Plenty of rules and plugins to use
    - Heritrix requires Java environment
  - Better heuristics:
    - Check for cookie manipulation
    - Check XSS also on: onLoad, onmouseover, img src etc.

OWASP
The Open Web Application Security Project

- Next Version
  - Django App for reporting via Grappelli
  - Support for Heritrix 3
    - WARC ingester directly to DB
    - API Restful interface for automation
  - Support for malware blacklists
  - Extend Thug plugins?
  - Plugin for Microsoft Zozzle?

OWASP
The Open Web Application Security Project

- Project will be on Github at some point this year
- Vagrant image will be on the cloud

- Contact me: Paolo.DiProdi@microsoft.com