Honeyclients

Low-interaction detection methods

Thijs Stuurman & Alex Verduin

06-02-2008

UNIVERSITEIT VAN AMSTERDAM

System and Network Engineering

Index

- Introduction
- Research question
- Research
- Interesting findings
- Pitfalls
- Conclusions
- Questions

Phishing example

Spam:

You have have won a free iPhone!

Research question

"Investigate how to determine that a webpage is suspicious of holding malicious web content and should be further examined."

What are Honeyclients?

- Honeypots
- Honeyclients
- High-interaction
- Low-interaction

- Benefits
- Drawback

Approach

Approach:

- Reading
- Data gathering
- Analyses
- Findings

Repositories

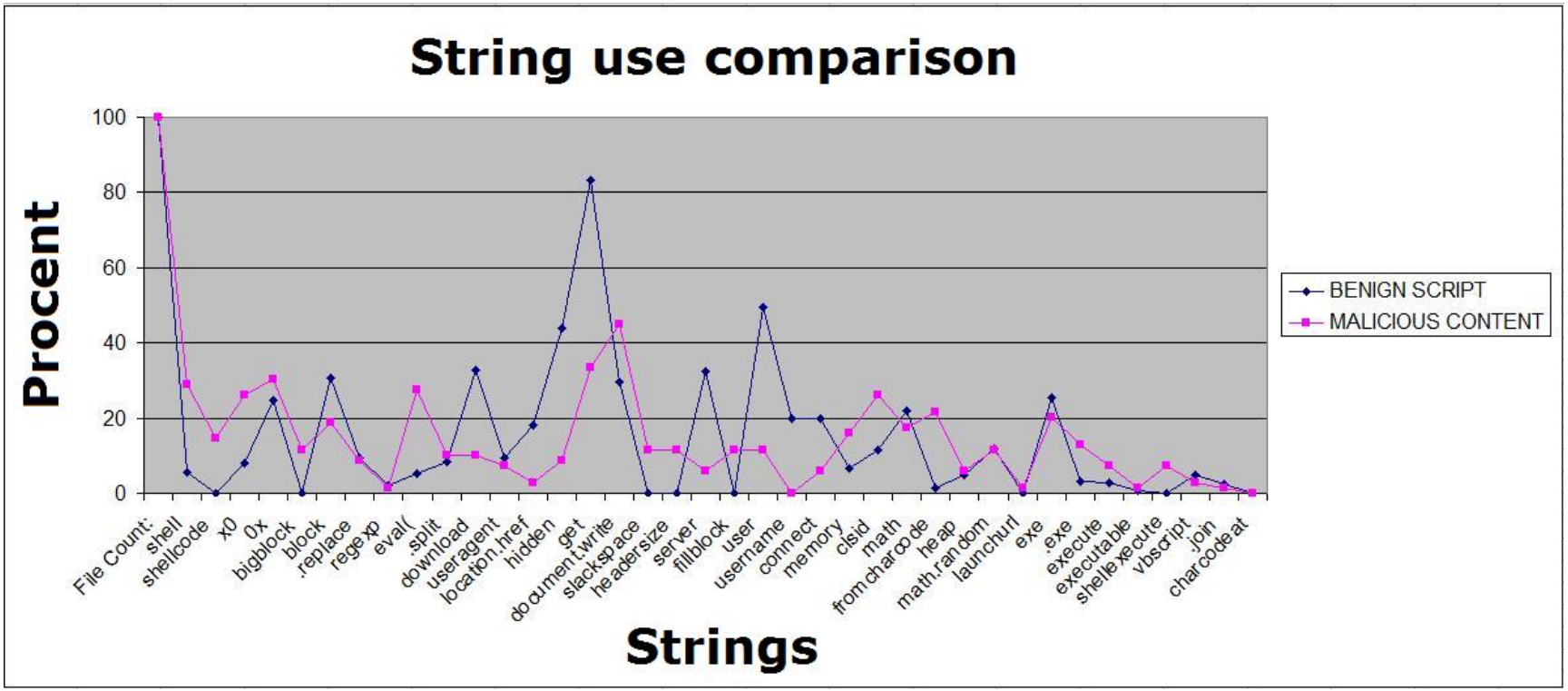
Data gathering:

- Capture
- URL lists
- Python, wget (user-agent, referer)

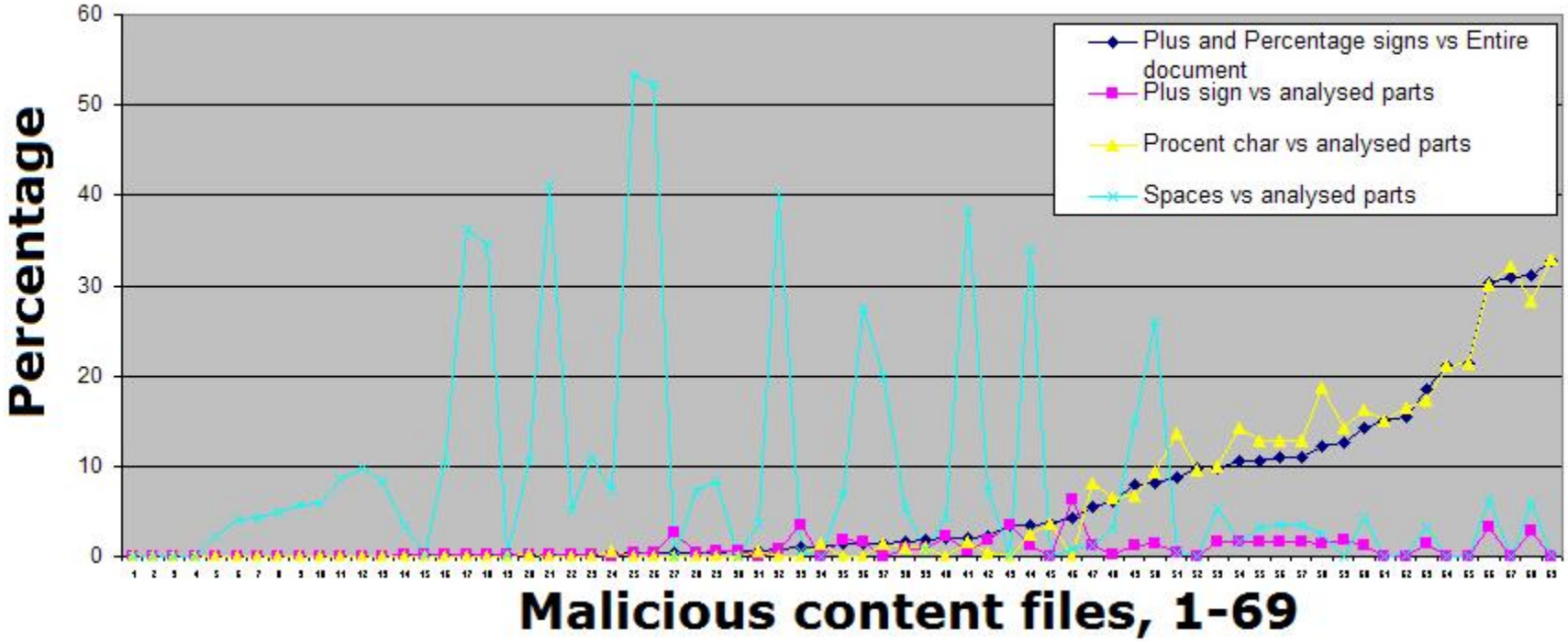
Malicious code

```
var shellcode = unescape("%uf3e9%u0000%u9000"+
"%u9090%u5a90%ua164%u0030%u0000%u408b%u8b0c" +
"%u1c70%u8bad%u0840%ud88b%u738b%u8b3c%u1e74%u0378" +
"%u8bf3%u207e%ufb03%u4e8b%u3314%u56ed%u5157%u3f8b" +
"%ufb03%uf28b%u0e6a%uf359%u74a6%u5908%u835f%u04c7" +
"%ue245%u59e9%u5e5f%ucd8b%u468b%u0324%ud1c3%u03e1" +
"%u33c1%u66c9%u088b%u468b%u031c%uc1c3%u02e1%uc103" +
"%u008b%uc303%ufa8b%uf78b%uc683%u8b0e%u6ad0%u5904" +
"%u6ae8%u0000%u8300%uf3ee%u5652%u57ff%u5afc%ud88b" +
"%u016a%ue859%u0057%u0000%uc683%u5613%u8046%u803e" +
"%ufa75%u3680%u5e80%uec83%u8b40%uc7dc%u6303%u646d" +
```

Findings 1/4


String use comparison
Procent
100
80
60
40
20
0
File Count:
shell
shellcode
x0
0x
bigblock
block
replace
regexp
eval(
.split
download
useragent
location.href
hidden
get
document.write
slackspace
headersize
server
fillblock
user
username
connect
memory
clsid
math
fromcharcode
heap
math.random
launchurl
exe
.exe
execute
executable
shellexecute
vbscript
join
charcodeat
Strings
BENIGN SCRIPT
MALICIOUS CONTENT

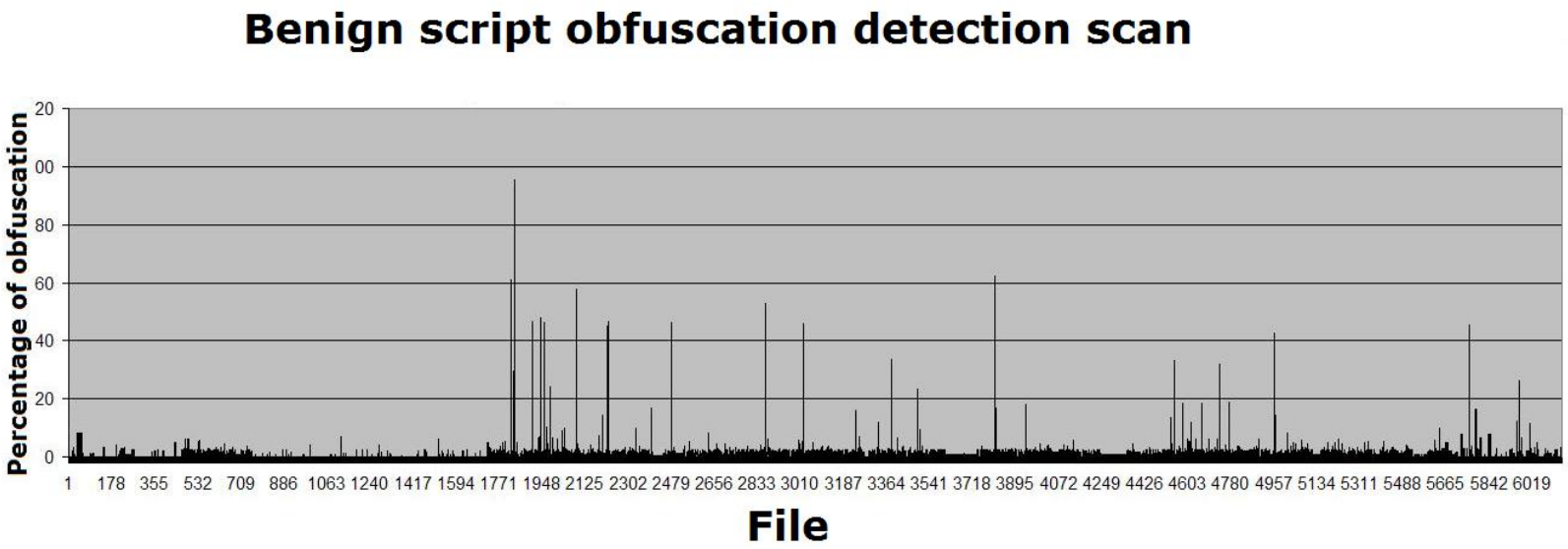
Findings 2/4



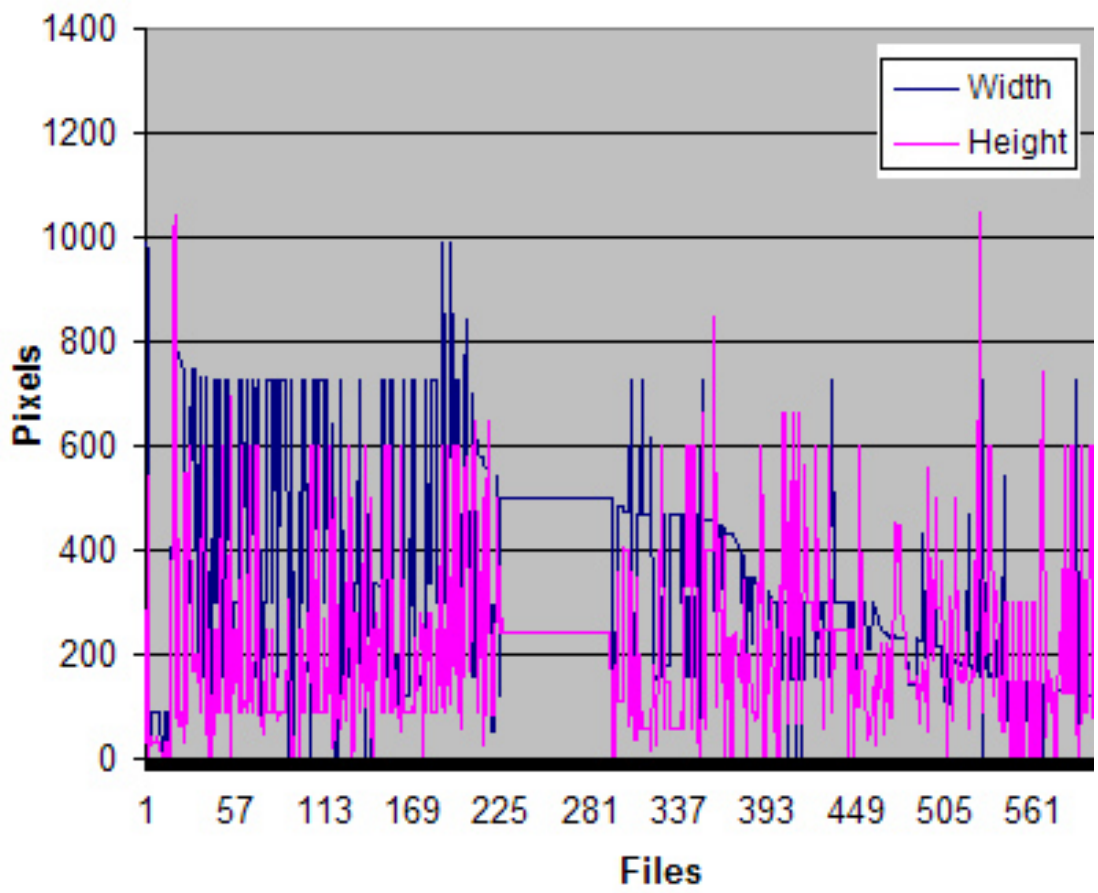
"var+var+var+var+var"

%6C%2E%69%6E%66%6F%2

Findings 3/4

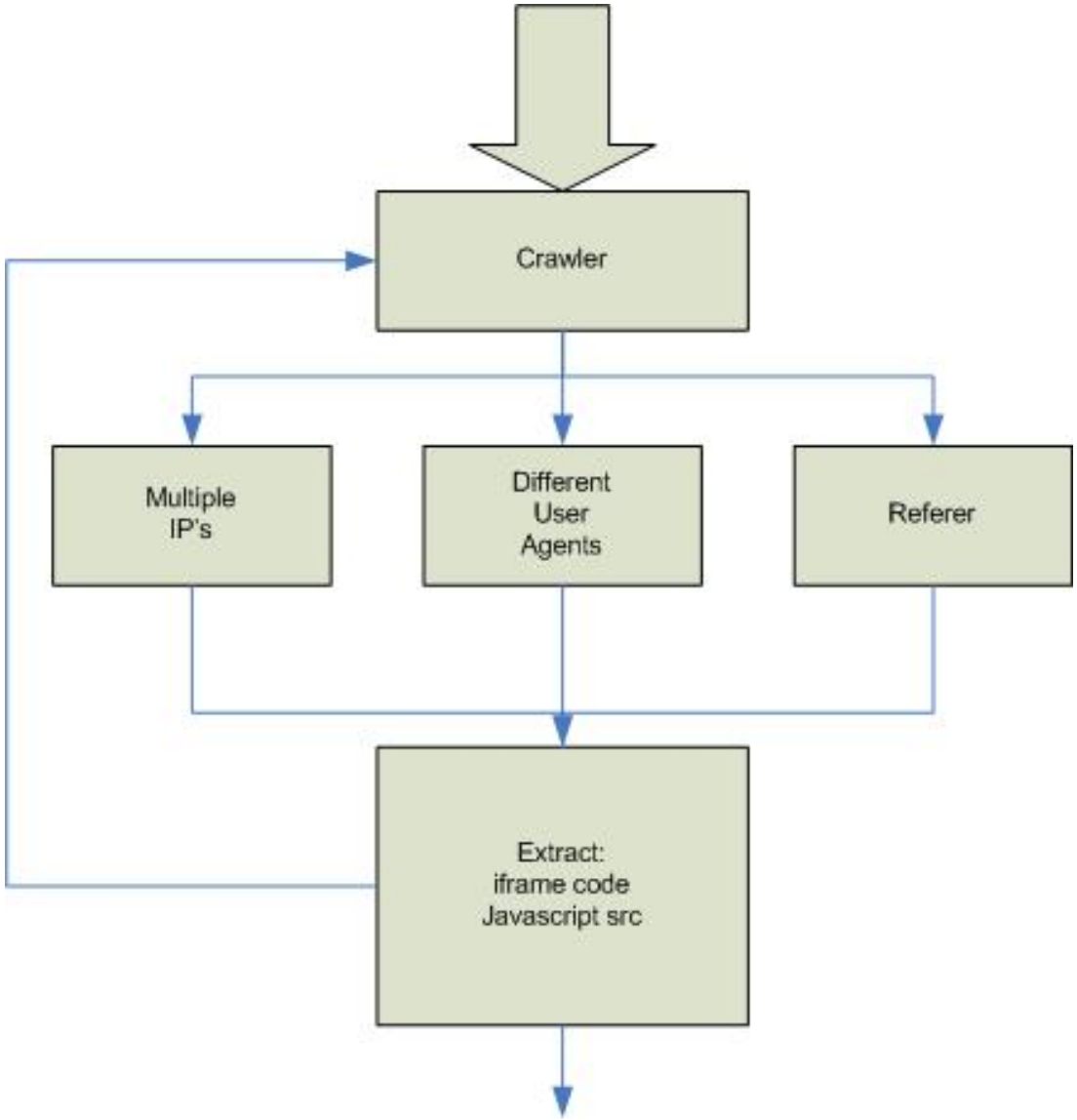

Benign script obfuscation detection scan
Percentage of obfuscation
20
00
80
60
40
20
0
1 178 355 532 709 886 1063 1240 1417 1594 1771 1948 2125 2302 2479 2656 2833 3010 3187 3364 3541 3718 3895 4072 4249 4426 4603 4780 4957 5134 5311 5488 5665 5842 6019
File

Findings 4/4

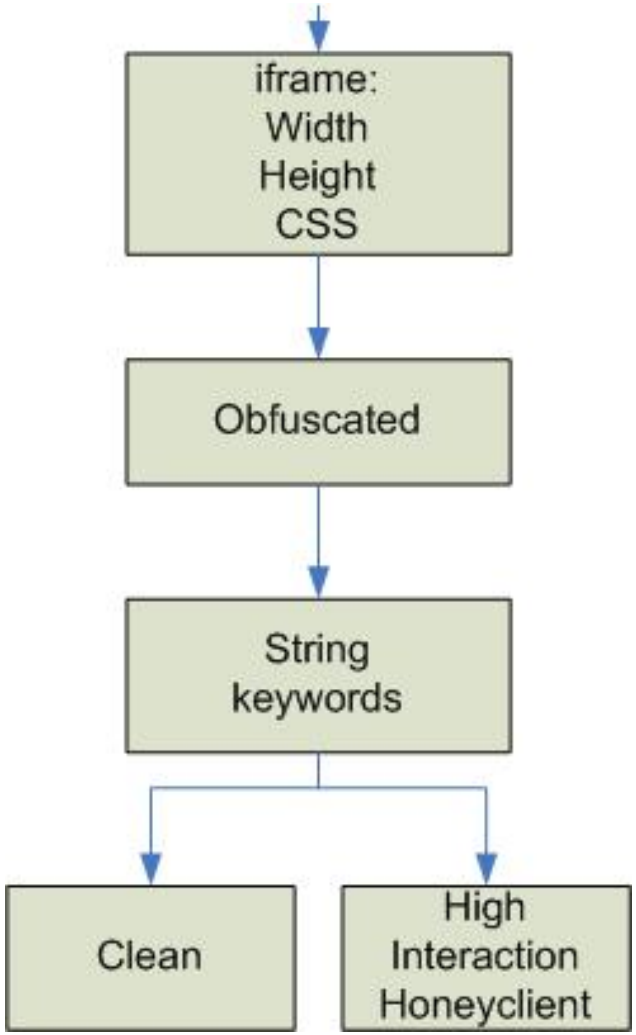



```
<iframe src="http://www.url.com/"
style="visibility:hidden;style:none;z-index:3;
top: 0px; left :0px;" width="1" height=1>
```

Detection model 1/2



Detection model 2/2



Pitfalls

Pitfalls:

- Unknown, creative obfuscation methods
- Serverside intelligence
- New exploits

Conclusion and further work

Conclusions:

- Basic detection model
- But there are still challenges
  - For example, resolve the pitfalls
- We expect future challenges

Future work:

- Research on a bigger data collection
- Advanced debugging JavaScript interpreter
- Improve our model
  - Define more keywords
  - Better analyse method for obfuscation

Questions

Questions?