Man vs Machine

Manual and automated security testing

About

- Security of web applications
- Assurance part of SDLC only
- Compare strengths and weaknesses of manual vs automated test
- Based on personal experiences mainly
- Manual tester (might be biased)

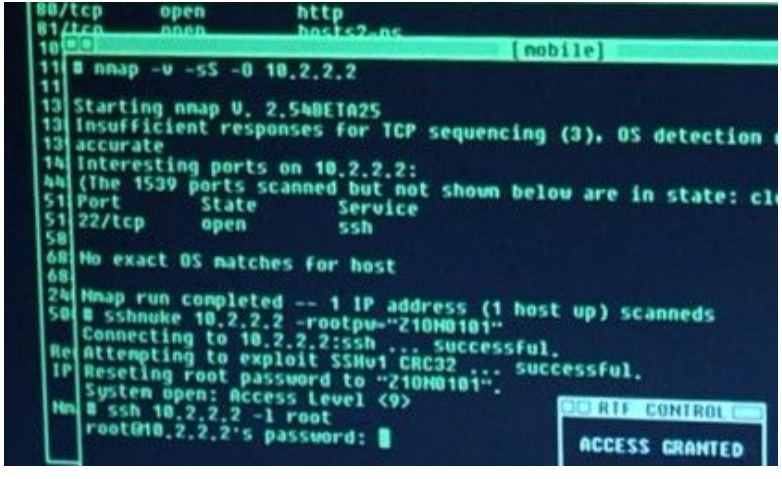
Two kingdoms of automation

SAST

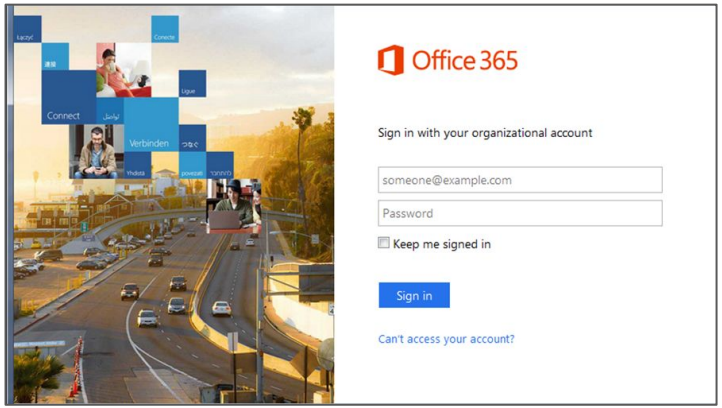
Static Application Security Testing

DAST

Dynamic Application Security Testing



From on-premise to SaaS



Transition of responsibility for security

Insane speed of release cycle

Security challenges shift to the application level

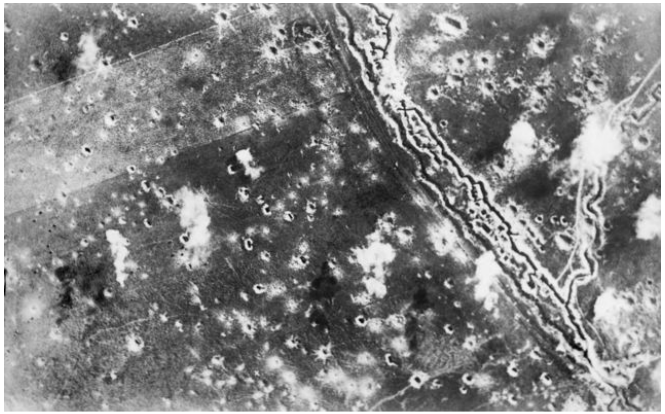
Scope coverage

Humans:

- Unreliable (!?)
- Scope creep
- Traverse through integrations
- A lot depends on the individualities

Machines:

- Narrow
- Reliable
- Difficulties with testing integrations
- Limited support of technologies



Speed

Humans:

- Slow test process
- Can start with new app immediately

Machines:

- Fast test process
- Time to onboard (days with SAST)

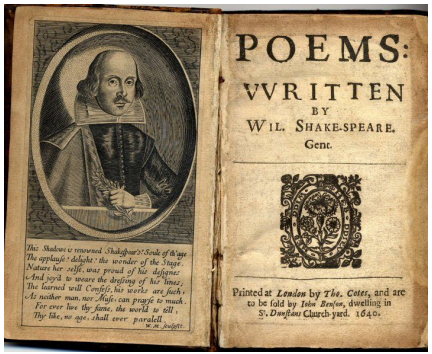
Reporting

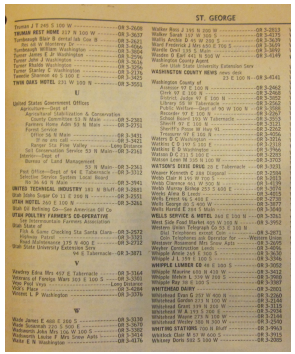
Humans:

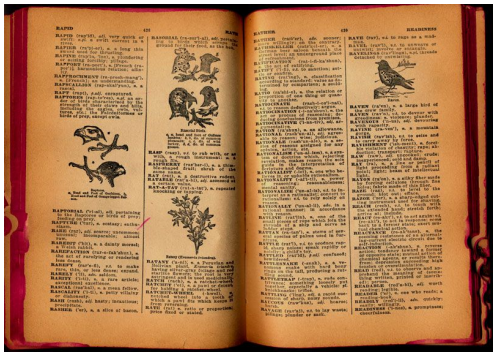
- No false positives
- Tend to group systemic findings
- Linked to app logic
- Insights about business impact

Machines:

- False positives
- Each vector as a separate finding
- Challenges @correlation/deduplication







Land lost to machines

Enumerate known badness:

- ✓ Missing infrastructure patches
- ✓ Outdated dependencies
- ✓ Known configuration issues

Garry Kasparov vs Deep Blue, 1997

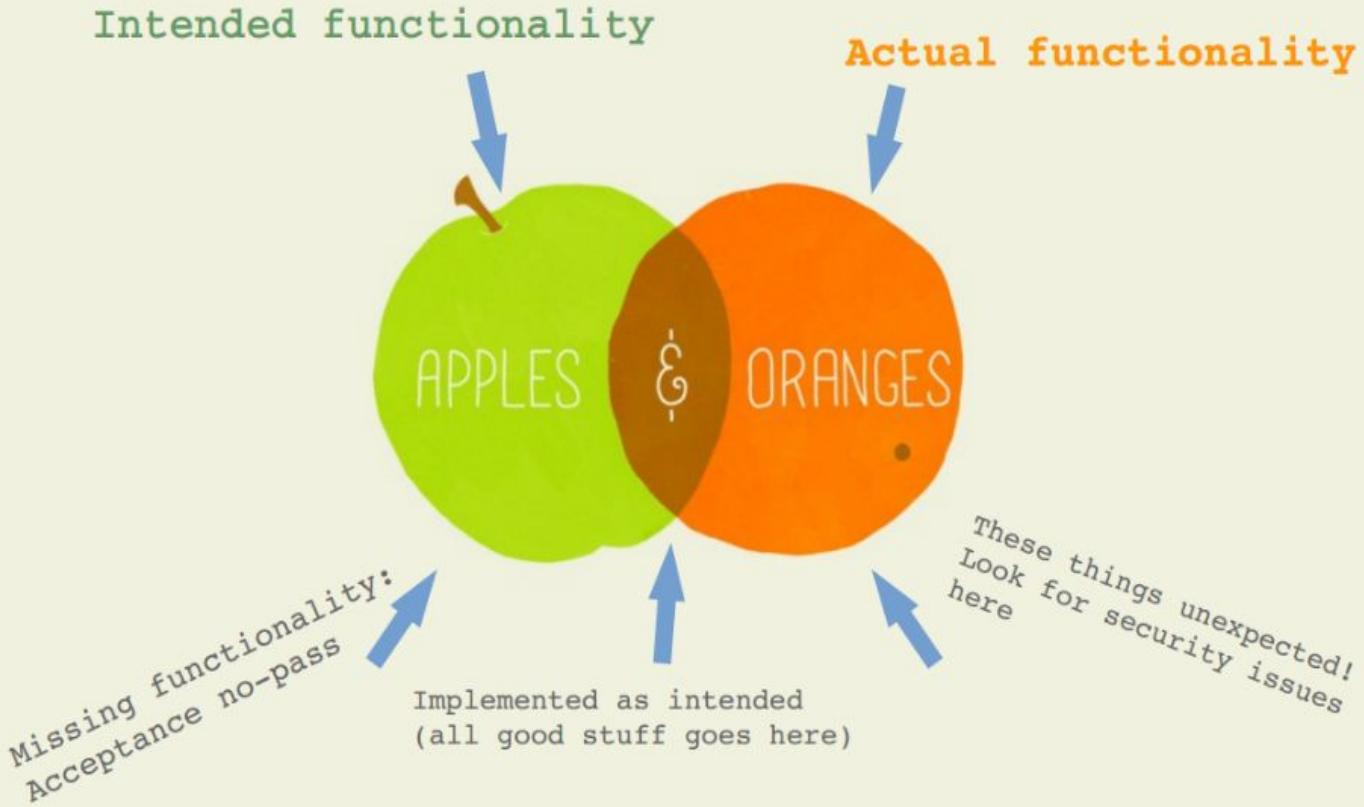
In the application layer

0-day every day!

Because

- Application bugs are custom
- Unexplored by researchers
- Apps are buggy!
  - ~40 XSS/app in average

Apples and oranges



Companies classify vulnerabilities as non-functional, while hackers see them as features that can be utilised in an attack.

OWASP Top 10 2017

The Ten Most Critical Web Application Security Risks

Release Candidate 2

Comments requested per instructions within

Analysis of OWASP Top 10 data sets

24 different contributors

SAST, DAST and manual testing

2.3 million vulnerabilities

55 034 applications

~42 vulnerabilities / application

Human-Augmented Tools (HAT) vs. Tool-Augmented Humans (TAH)

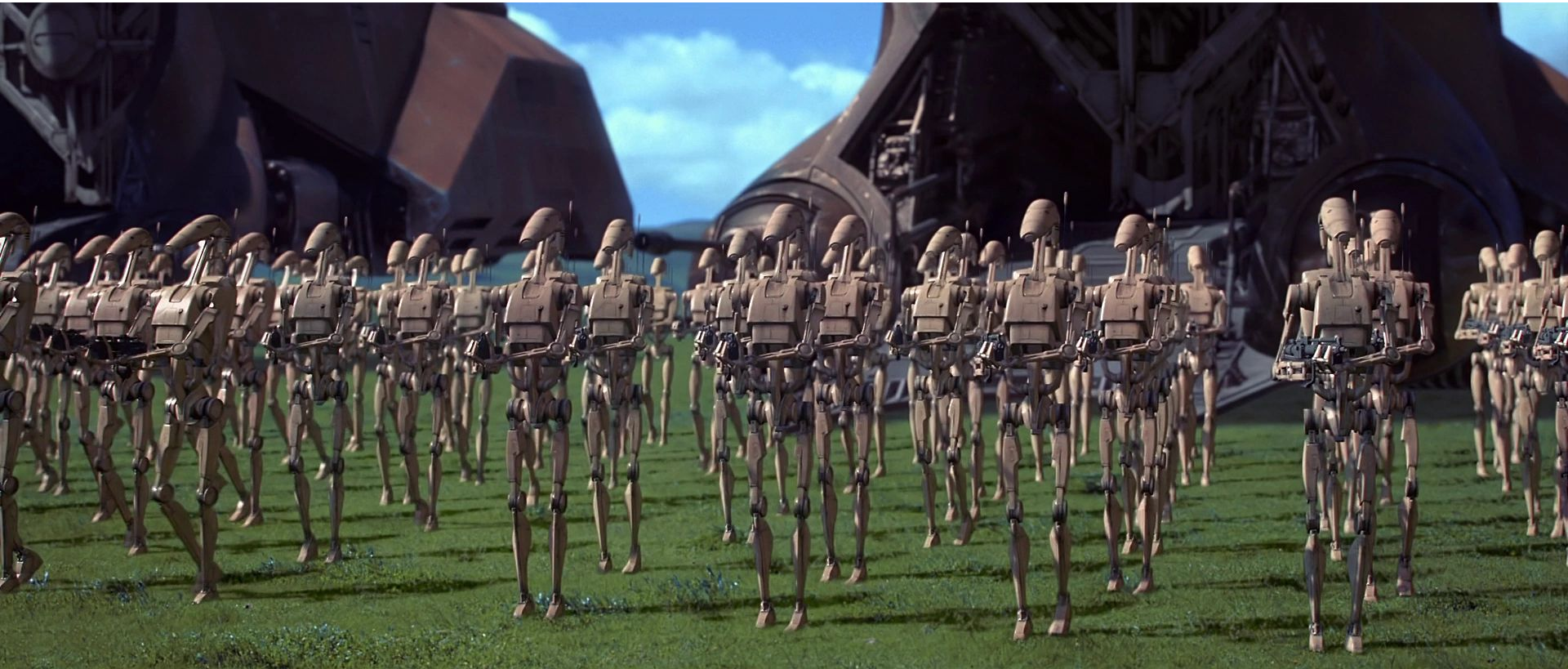
91% of applications tested by HAT

Complete analysis: https://nvisium.com/blog/2017/04/18/musings-on-the-owasp-top-10-2017-rc1/

Vulnerabilities	Totals	Human %	Machine %
Web Applications	55034	9.1%	90.9%
Number of SQL Injection Vulnerabilities Found (CWE-89)?	183217	0.7%	99.3%
Number of Hibernate Injection Vulnerabilities Found (CWE-564)?	2096	0.7%	99.3%
Number of Command Injection Vulnerabilities Found (CWE-77)?	8086	2.2%	97.8%
Number of Authentication Vulnerabilities Found (CWE-287)?	9004	58.7%	41.3%
Number of Session Fixation Vulnerabilities Found (CWE-384)?	4904	24.0%	76.0%
Number of Cross-Site Scripting (XSS) Vulnerabilities Found (CWE-79)?	1925226	0.3%	99.7%
Number of DOM-Based XSS Vulnerabilities Found (No CWE)?	350	66.0%	34.0%
Number of Insecure Direct Object Reference Vulnerabilities Found (CWE-639)?	4390	19.2%	80.8%
Number of Path Traversal Vulnerabilities Found (CWE-22)?	12489	4.9%	95.1%
Number of Missing Authorization Vulnerabilities Found (CWE-285)?	4069	55.5%	44.5%
Number of Security Misconfiguration Vulnerabilities Found (CWE-2)?	19225	43.3%	56.7%
Number of Cleartext Transmission of Sensitive Information Vulnerabilities Found (CWE-319)?	2844	69.9%	30.1%
Number of Cleartext Storage of Sensitive Information Vulnerabilities Found (CWE-312)?	1872	39.5%	60.5%
Number of Cryptographic Vulnerabilities Found (CWEs-310/326/327/etc)?	9831	11.3%	88.7%
Number of Improper (Function Level) Access Control Vulnerabilities Found (CWE-285)?	1411	92.7%	7.3%
Number of Cross-Site Request Forgery (CSRF) Vulnerabilities Found (CWE-352)?	1893	70.5%	29.5%
Number of Use of Known Libraries Found (NEW 937)?	33406	2.5%	97.5%
Number of Unchecked Redirect Vulnerabilities Found (CWE-601)?	57459	0.6%	99.4%
Number of Unvalidated Forward Vulnerabilities Found (No CWE)?	919	14.8%	85.2%
Number of Clickjacking Vulnerabilities Found (No CWE)?	4269	47.2%	52.8%
Number of XML eXternal Entity Injection (XXE) Vulnerabilities Found (CWE-611)?	42387	0.6%	99.4%
Number of Server-Side Request Forgery (SSRF) Vulnerabilities Found (CWE-918)?	229	2.2%	97.8%
Number of Denial of Service (DOS) Vulnerabilities Found (CWE-400)?	1563	83.3%	16.7%
Number of Expression Language Injection Vulnerabilities Found (CWE-917)?	81	56.8%	43.2%
Number of Error Handling Vulnerabilities Found (CWE-388)?	4848	47.3%	52.7%
Number of Information Leakage/Disclosure Vulnerabilities Found (CWE-200)?	6088	42.5%	57.5%
Number of Insufficient Anti-automation Vulnerabilities Found (CWE-799)?	842	85.2%	14.8%
Number of Insufficient Security Logging Vulnerabilities Found (CWE-778)?	1051	43.1%	56.9%
Number of Insufficient Intrusion Detection and Response Vulnerabilities Found (No CWE)?	69	49.3%	50.7%
Number of Mass Assignment Vulnerabilities Found (CWE-915)?	5171	2.3%	97.7%
Input Validation	4699	0.0%	100.0%
Unrestricted Upload of File with Dangerous Type (CWE-434)	14	100.0%	0.0%
Totals:	2354002	1.8%	98.2%

Have the humans complete 10x apps			
Machine %	Human %	Totals	Vulnerabilities
50.0%	50.0%	99989	Web Applications
93.7%	6.3%	194296	Number of SQL Injection Vulnerabilities Found (CWE-89)?
93.7%	6.3%	2222	Number of Hibernate Injection Vulnerabilities Found (CWE-564)?
81.7%	18.3%	9679	Number of Command Injection Vulnerabilities Found (CWE-77)?
6.6%	93.4%	56596	Number of Authentication Vulnerabilities Found (CWE-287)?
24.0%	76.0%	15497	Number of Session Fixation Vulnerabilities Found (CWE-384)?
97.5%	2.5%	1969794	Number of Cross-Site Scripting (XSS) Vulnerabilities Found (CWE-79)?
4.9%	95.1%	2429	Number of DOM-Based XSS Vulnerabilities Found (No CWE)?
29.7%	70.3%	11959	Number of Insecure Direct Object Reference Vulnerabilities Found (CWE-639)?
66.0%	34.0%	17988	Number of Path Traversal Vulnerabilities Found (CWE-22)?
7.4%	92.6%	24409	Number of Missing Authorization Vulnerabilities Found (CWE-285)?
11.6%	88.4%	94078	Number of Security Misconfiguration Vulnerabilities Found (CWE-2)?
4.1%	95.9%	20736	Number of Cleartext Transmission of Sensitive Information Vulnerabilities Found (CWE-319)?
13.3%	86.7%	8523	Number of Cleartext Storage of Sensitive Information Vulnerabilities Found (CWE-312)?
43.9%	56.1%	19839	Number of Cryptographic Vulnerabilities Found (CWEs-310/326/327/etc)?
0.8%	99.2%	13183	Number of Improper (Function Level) Access Control Vulnerabilities Found (CWE-285)?
4.0%	96.0%	13908	Number of Cross-Site Request Forgery (CSRF) Vulnerabilities Found (CWE-352)?
79.4%	20.6%	41029	Number of Use of Known Libraries Found (NEW 937)?
94.1%	5.9%	60699	Number of Unchecked Redirect Vulnerabilities Found (CWE-601)?
36.5%	63.5%	2143	Number of Unvalidated Forward Vulnerabilities Found (No CWE)?
10.1%	89.9%	22395	Number of Clickjacking Vulnerabilities Found (No CWE)?
94.4%	5.6%	44628	Number of XML eXternal Entity Injection (XXE) Vulnerabilities Found (CWE-611)?
81.8%	18.2%	274	Number of Server-Side Request Forgery (SSRF) Vulnerabilities Found (CWE-918)?
2.0%	98.0%	13281	Number of Denial of Service (DOS) Vulnerabilities Found (CWE-400)?
7.1%	92.9%	495	Number of Expression Language Injection Vulnerabilities Found (CWE-917)?
10.0%	90.0%	25476	Number of Error Handling Vulnerabilities Found (CWE-388)?
11.9%	88.1%	29362	Number of Information Leakage/Disclosure Vulnerabilities Found (CWE-200)?
1.7%	98.3%	7295	Number of Insufficient Anti-automation Vulnerabilities Found (CWE-799)?
11.7%	88.3%	5128	Number of Insufficient Security Logging Vulnerabilities Found (CWE-778)?
9.3%	90.7%	375	Number of Insufficient Intrusion Detection and Response Vulnerabilities Found (No CWE)?
81.1%	18.9%	6233	Number of Mass Assignment Vulnerabilities Found (CWE-915)?
100.0%	0.0%	4699	Input Validation
0.0%	100.0%	140	Unrestricted Upload of File with Dangerous Type (CWE-434)
84.4%	15.6%	2738788	Totals:

Sad but obvious: humans don’t scale



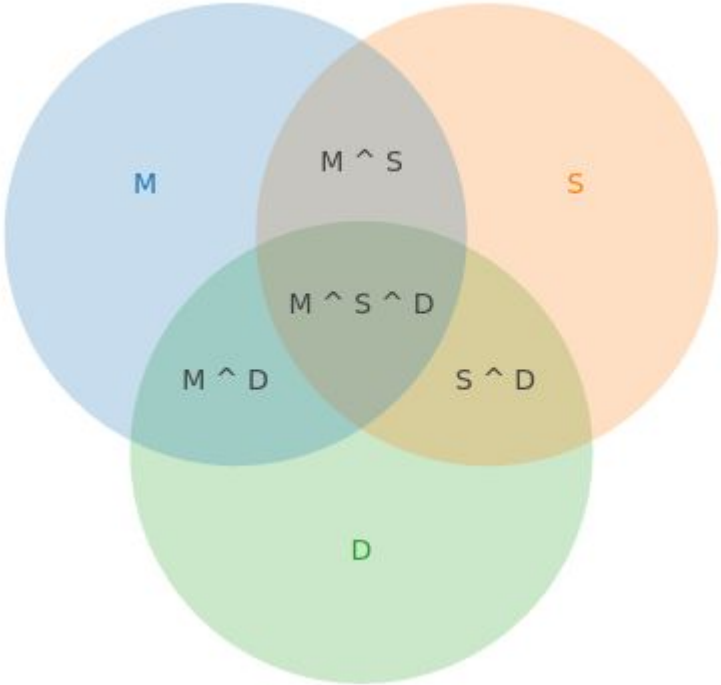
Defender’s dilemma

An attacker only needs to find one weakness while the defender needs to find every one.

Nakatomi space

Nakatomi space

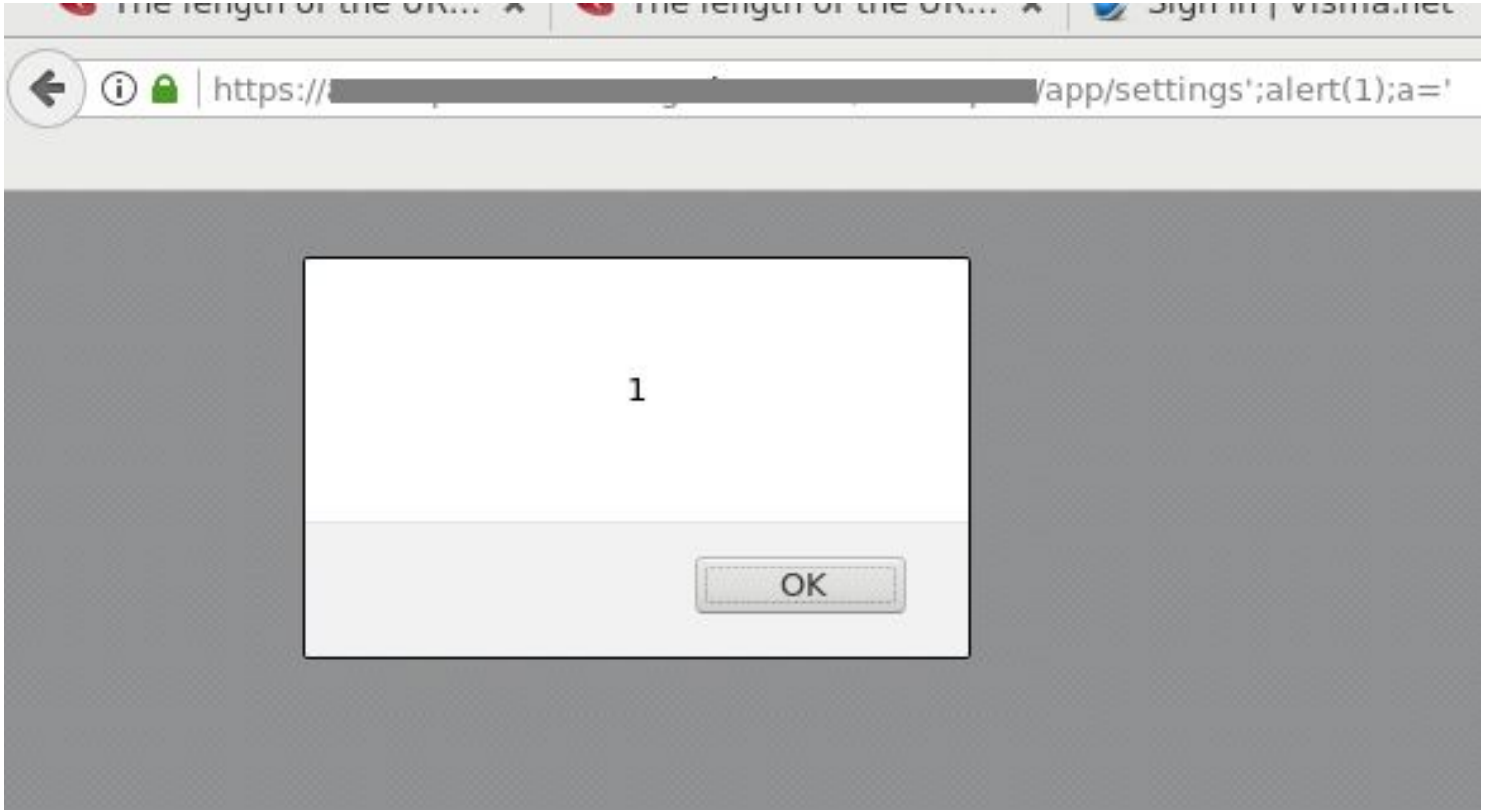
Overlap expected



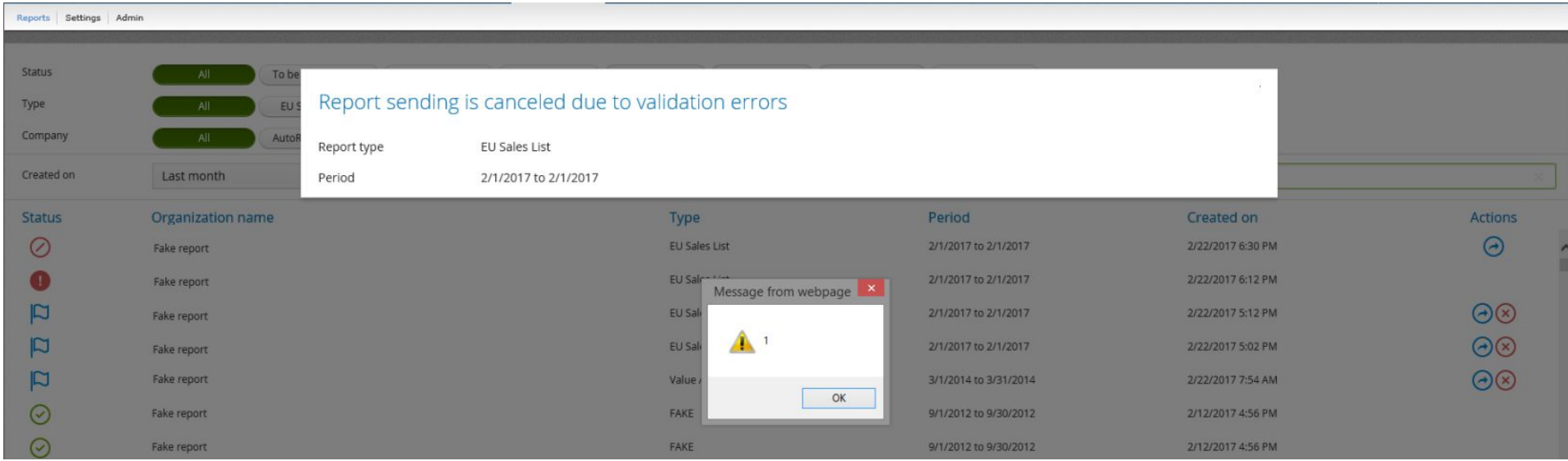
XSS in AppX (1): Manual test

```
GET /[redacted]/app/settings';alert(1);a=' HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:51.0) Gecko/20100101 Firefox/51.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: close
Upgrade-Insecure-Requests: 1
Pragma: no-cache
Cache-Control: no-cache
```

```
<html>
<body>
<script>try {
window.parent.location.href = 'https://[redacted]/loginwebapp/
loginPage.xhtml?requestedPage=https://[redacted]/
[redacted]/app/settings';alert(1);a='';
}
catch(SecurityError) {
window.location.href = 'https://[redacted]/loginwebapp/
loginPage.xhtml?requestedPage=https://[redacted]/
[redacted]/app/settings';alert(1);a='';
}</script>
```


https://[redacted]/app/settings';alert(1);a='
1
OK

XSS in AppX (2): Manual test



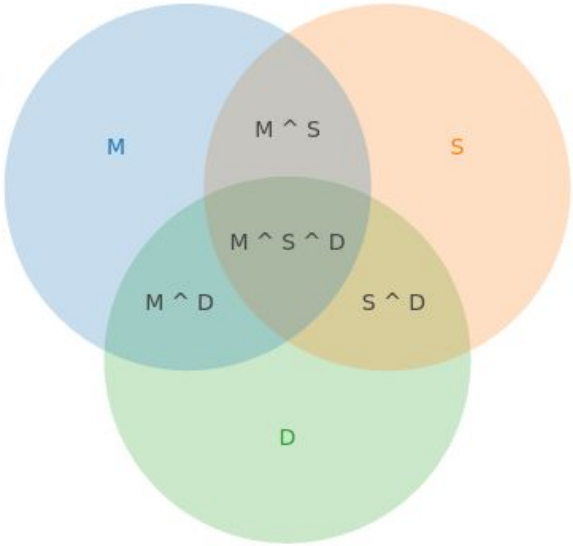
XSS in AppX (x2): SAST

```
@if (Model.Count > 0){
    <div class="filter singleSelect row" id="@ViewBag.Id">
        <label for="@ViewBag.Id" class="control-label col-xs-1 small-label">@ViewBag.FilterName</label>
        <div class="col-xs-11">
            <ul class="nav nav-pills nav-pills-primary">
                @foreach (KeyValuePair<string, string> item in Model)
                {
                    <li class="filterItem filterButton" data-value="@item.Key">
                        <a class="description">
                            @item.Value
                        </a>
                    </li>
                }
            </ul>
        </div>
    </div>
}
```

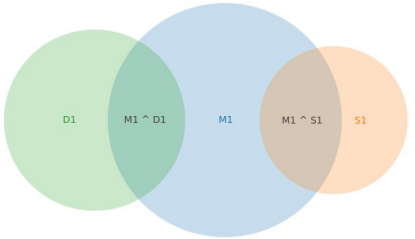
2 != 2

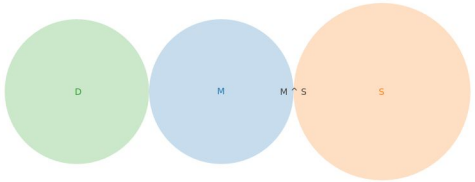
Overlap: Expected vs Actual

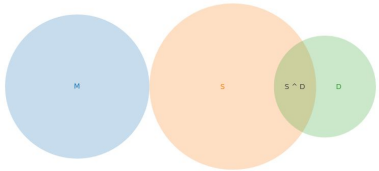
Expected:



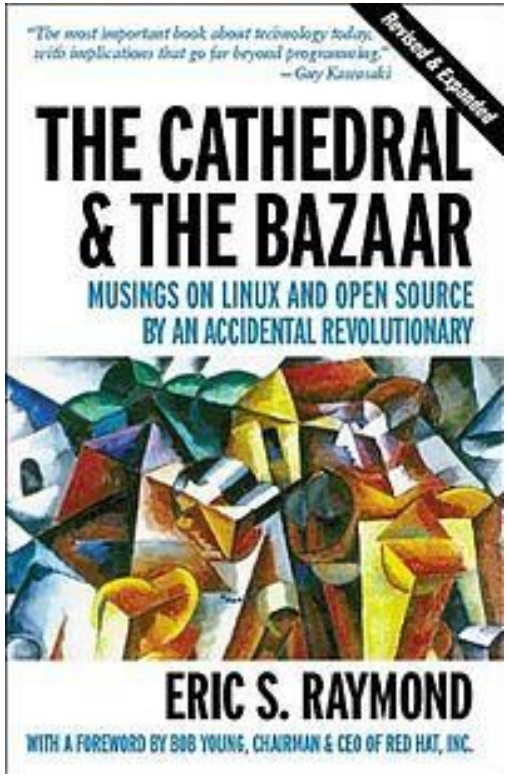
Reality:







Given enough eyeballs..

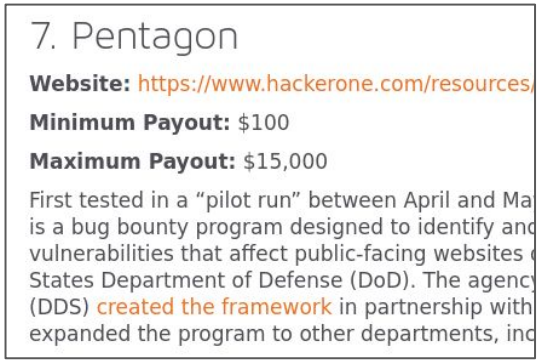


Bugbounty

Platform providers: hackerone Synack bugcrowd ...

Testers: Anyone!

Rewards: 50 $ .. >5 000 $ (current max **@h1** is 30 000 $)



Black market of web bugs

Actor Profile: Yummba

"Yummba" is a highly proficient, Russian-speaking hacker and author of the infamous ATS web injects, which targeted multiple financial organization all over the world and caused damage estimated at tens of millions of dollars.

Yummba develops highly customized tools, tailored specifically for each customer. (...) significantly more expensive than tools created by other developers, and command **prices upwards of $1,000**. Typically Yummba's **web-injects** include full source code, and buyers are allowed to resell it at any time.

Yummba's software is more powerful than its analogs because of their ATS Engine web injects, which not only compromise a client device or network, but portions of these attacks **might also be used in cross-site scripting, phishing, and drive-by download attacks.**

Buy your bugs back!

… before criminals will do

Unique strengths of humans & machines

No False positives

Context aware

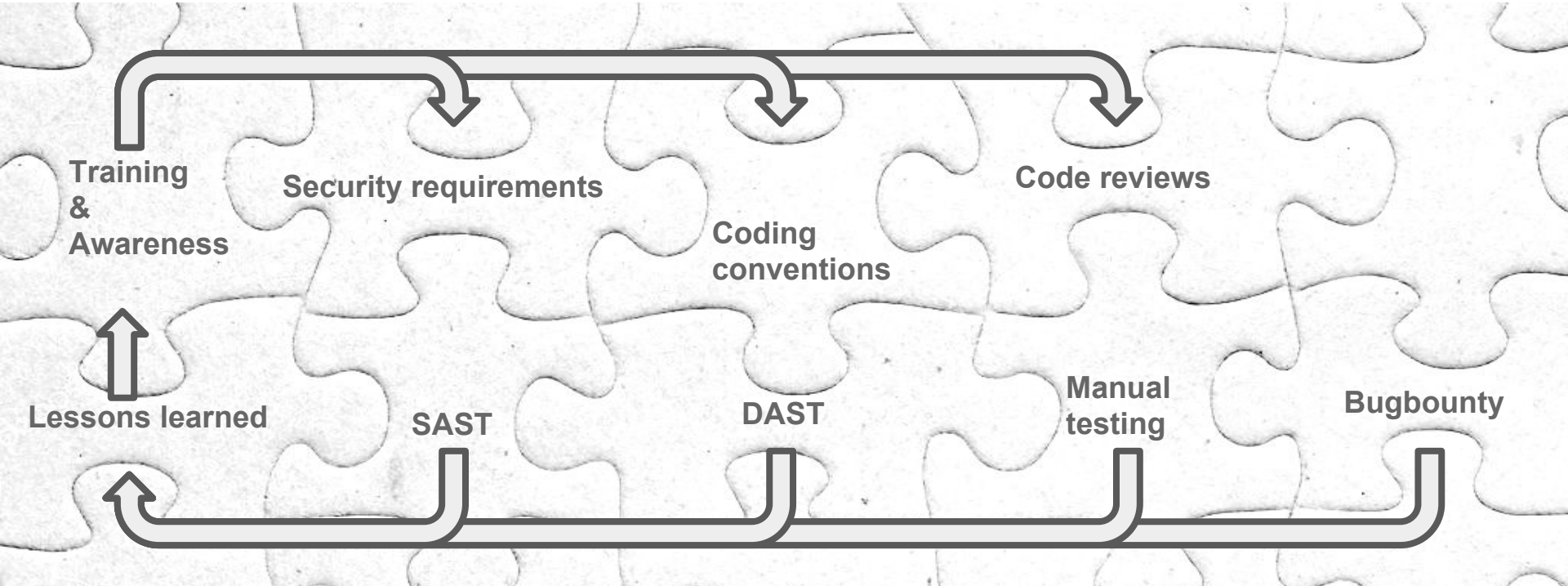
Nakatomi space

Fast

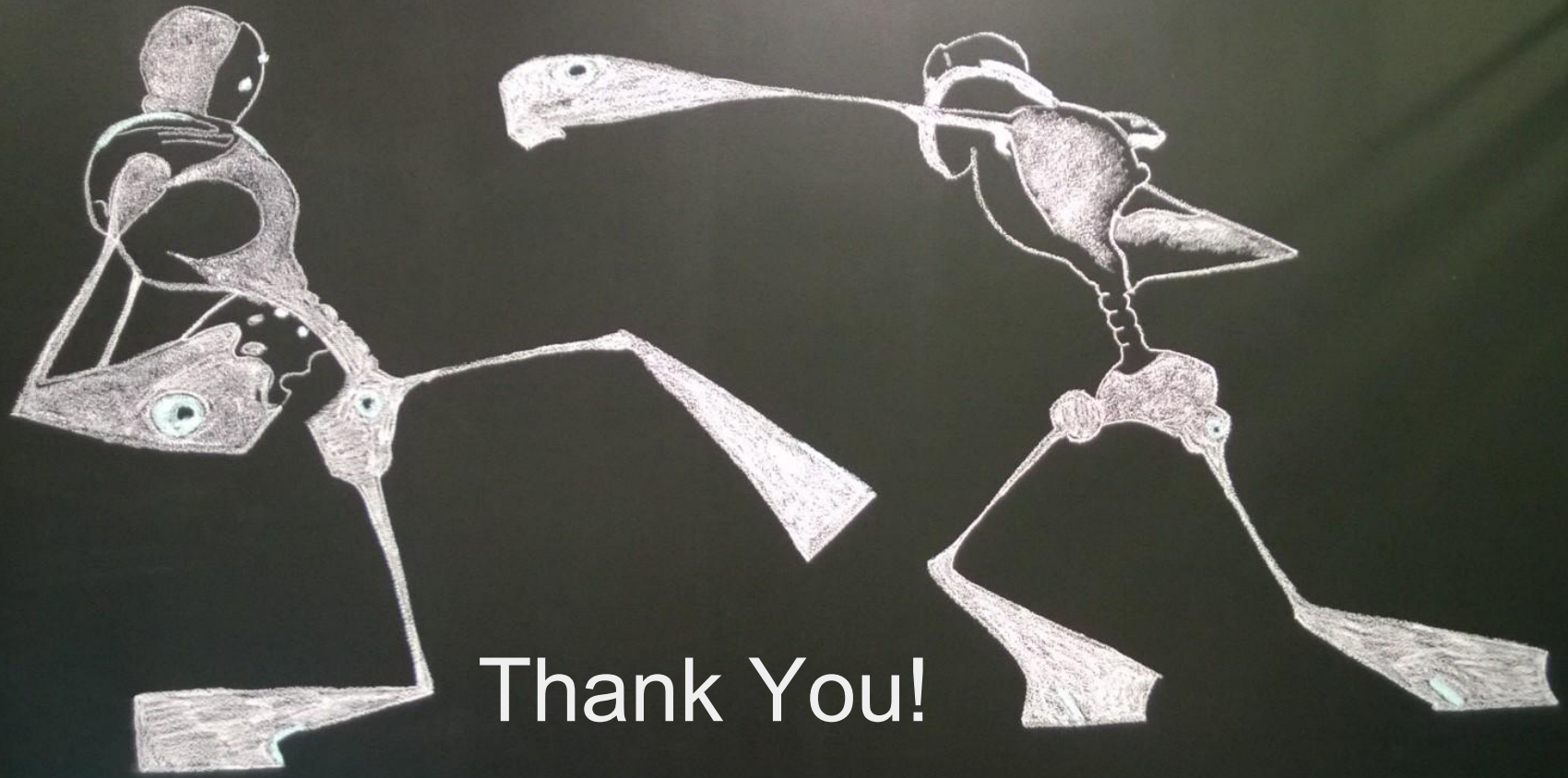
Scalable

Repetitive tasks

The complete picture



Q&A



Thank You!

aurelijus.stanislovaitis@visma.com