Open Universiteit

HLISA*: a Human-Like Simulation API for webpage interaction

Hugo Jonker
with Benjamin Krumnow, David Roefs, Daniel Goßen, Stefan Karsch

** pronounced "hey-lisa"*

But first:

New publication!

Download link:

https://authors.elsevier.com/c/1drCQc43uuxhG

Coincidentally: it’s relevant :)

COMPUTERS & SECURITY 111 (2021) 102472

ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Measuring Web Session Security at Scale☆

Stefano Calzavara[a], *Hugo Jonker*[b,c], *Benjamin Krumnow*[b,d,*], *Alvise Rabitti*[a]

[a] *Università Ca’ Foscari Venezia, Italy*
[b] *Open University of the Netherlands, Netherlands*
[c] *Radboud University Nijmegen, Netherlands*
[d] *TH Köln, Steinmüllerallee 1, Gummersbach 51643, Germany*

ARTICLE INFO

Article history:
Received 17 February 2021
Revised 8 July 2021
Accepted 12 September 2021
Available online 16 September 2021

MSC:
68M25

Keywords:
Session security
Shepherd
Black-box testing
Web measurements
Automated login
Authentication

ABSTRACT

Session management is a particularly delicate component of web applications, which might suffer from a range of severe security issues, including impersonation attacks. Unfortunately, the scope and significance of prior work on web session security in the wild are limited by the complexity of the attack surface and the challenges of automating the login process on existing websites. In the present article, we fill this gap by proposing the first comprehensive, large-scale web session security measurement based on *post-login* data. Our analysis is comprehensive in that it deals with all key aspects of web sessions, i.e., the login process, the logout process and the authentication cookie handling. Our automated approach analysed an extensive set of session management practices of over 6,000 sites where login was successful and authentication cookies could be automatically detected, uncovering a widespread adoption of insecure practices in the wild.

TL;DR “Measuring web security”

- We **automatically** login on 6,124 sites and measure security
-

Table 2 – Login security results by site popularity.

Site popularity	≤1M	
Successful logins	6,124	100%
Password theft	909	15%
– login form sent over HTTP	755	12%
– login page served over HTTP	901	15%
– password in query string	4	0%
Password brute-forcing	5,347	87%

Table 4 – Cookie security results by site popularity.

Site popularity	≤1M	
Successful logins	6,124	100%
Session hijacking via network sniffing	1,398	23%
Session hijacking via JavaScript	2494	41%
Session fixation	1,011	16%
Cookie brute-forcing	2,044	33%
– weak session identifiers in cookies	1,981	32%
– weak password hashes in cookies	63	1%

Table 6 – Session invalidation results by site popularity.

	≤1M	
Logged out	3,302	100%
Server-side invalidation:	2,833	86%
– immediately	2,601	79%
– within 5 minutes	97	3%
– 5 minutes – 10 days	135	4%
– unknown, > 10 days	469	14%
Client-side left PII behind in:	230	7%
– localStorage	48	2%
– Cookies_{loc}	199	6%
– Cookies_{net}	186	6%

In other words...

We found plenty of...

- ...login forms that are insecure
- ...sites that accept (/ require) weak passwords
- ...session cookies which can be stolen
- ...session identifiers which can be fixated
- ...session identifiers wchich aren’t cleaned up (client-side)
- ...sessions which aren’t invalidated (server-side)

In other words...

We found plenty of...

- ...login forms that are insecure
- ...sites that accept (/ require) weak passwords
- ...session cookies which can be stolen
- ...session identifiers which can be fixated
- ...session identifiers wchich aren't cleaned up (client-side)
- ...sessions which aren't invalidated (server-side)

Did we measure the right thing?

Which internet did we measure?

Is there a difference?

[JKV19]: →

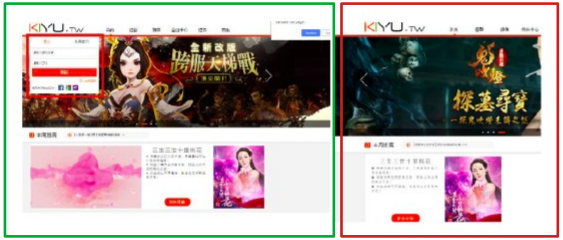


Fig. 4. Missing login fields on kiyu.tw.

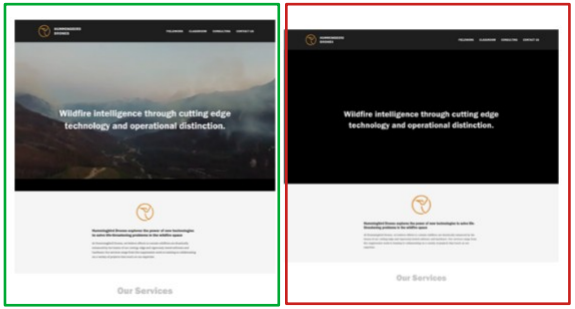


Fig. 5. Missing video on hummingbirddrones.ca.

Fig. 6. Blockage and loading of a CAPTCHA on frankmotorsinc.com.

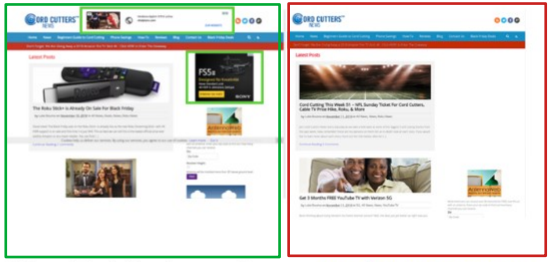


Fig. 7. Missing ads on cordcuttersnews.com.

Is our study doomed?

- No: would require faking logins
  - But extrapolation is hindered even further

Is our study doomed?

- No: would require faking logins
  - But extrapolation is hindered even further

- What we need:
  - a web bot that can fool the detectors

Is our study doomed?

- No: would require faking logins
  - But extrapolation is hindered even further

- What we need:
  - a web bot that can fool the detectors

Bot detectors in “The Terminator”

```
Welcome to
                    EEEEEE  LL      IIII  ZZZZZZZ   AAAAA
                    EE      LL       II       ZZ   AA   AA
                    EEEEE   LL       II      ZZZ   AAAAAAA
                    EE      LL       II     ZZ     AA   AA
                    EEEEEE  LLLLLL  IIII  ZZZZZZZ  AA   AA

Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.
```

HLISA: Human-Like Interaction Selenium Api

```
ELIZA: Please tell me what's been bothering you.
YOU:   I don't really know
ELIZA: Do you wish to be able to really know ?
YOU:   Hmm.. well, I guess yes
ELIZA: Do you say you guess yes for some special reason ?
YOU:   I don't really know
ELIZA: Does that trouble you ?
YOU:   What do you think?
ELIZA: What are your feelings now ?
YOU:   Sad
```

How to fool detectors

1. “Look like a human”:
   Avoid obvious points of detection

2. Behave like a human

1. Obvious points of detection

How to make those changes?

(1) **defineProperty** is a built-in function of JavaScript objects to directly set or alter an object's property.
(2) **__defineGetter__** overrides a `getter`-function allowing us to return a specific value without changing it. Note that this function was deprecated by Mozilla.
(3) **setPrototypeOf** sets a new prototype for an object, which provides control the access to properties.
(4) **Proxy objects** allow to re-define the behaviour of an object via wrapping it a with the proxy object.

How good are those approaches?

(1) **defineProperty** is a built-in function of JavaScript objects to directly set or alter an object's property.
(2) **__defineGetter__** overrides a `getter`-function allowing us to return a specific value without changing it. Note that this function was deprecated by Mozilla.
(3) **setPrototypeOf** sets a new prototype for an object, which provides control the access to properties.
(4) **Proxy objects** allow to re-define the behaviour of an object via wrapping it a with the proxy object.

Table 1: Detectable side effects by spoofing methods

Side effect	Spoofing method 1	2	3	4
Incorrect order of `navigator` properties	×	×		
Modified `navigator._length`	×	×		
New `Object.keys(navigator)`	×	×		
Defined `navigator.__proto__.webdriver`			×	
Unnamed `window.navigator` functions				×

How good are those approaches?

(1) **defineProperty** is a built-in function of JavaScript objects to directly set or alter an object's property.
(2) **__defineGetter__** overrides a `getter`-function allowing us to return a specific value without changing it. Note that this function was deprecated by Mozilla.
(3) **setPrototypeOf** sets a new prototype for an object, which provides control the access to properties.
(4) **Proxy objects** allow to re-define the behaviour of an object via wrapping it a with the proxy object.

Table 1: Detectable side effects by spoofing methods

Side effect	Spoofing method 1	2	3	4
Incorrect order of `navigator` properties	×	×		
Modified `navigator._length`	×	×		
New `Object.keys(navigator)`	×	×		
Defined `navigator.__proto__.webdriver`			×	
Unnamed `window.navigator` functions				×

Why not 3?

- setPrototypeOf() needs to have `__proto__` defined…
  - …but it isn’t in regular Firefox

-

Why not 3?

- setPrototypeOf() needs to have `__proto__` defined...
  - ...but it isn't in regular Firefox

- Conversely, proxy objects require parsing to detect

```
//Call of a toString function of a built-in method
window.navigator.toString.toString();

// Output in a regular Firefox browser
  "function toString() {
     [native code]
   }"

// Output after shadowing methods via proxy objects
  "function () {
    [native code]
   }"
```

Why not 3?

- setPrototypeOf() needs to have `__proto__` defined...
  - ...but it isn't in regular Firefox

- Conversely, proxy objects require parsing to detect

```
//Call of a toString function of a built-in method
window.navigator.toString.toString();

// Output in a regular Firefox browser
  "function toString() {
     [native code]
   }"

// Output after shadowing methods via proxy objects
  "function () {
    [native code]
   }"
```

expected

result

Why not 3?

- setPrototypeOf() needs to have `__proto__` defined...
  - ...but it isn't in regular Firefox

- Conversely, proxy objects require parsing to detect

```
//Call of a toString function of a built-in method
window.navigator.toString.toString();

// Output in a regular Firefox browser
  "function toString() {
     [native code]
   }"

// Output after shadowing methods via proxy objects
  "function () {
    [native code]
   }"
```

expected

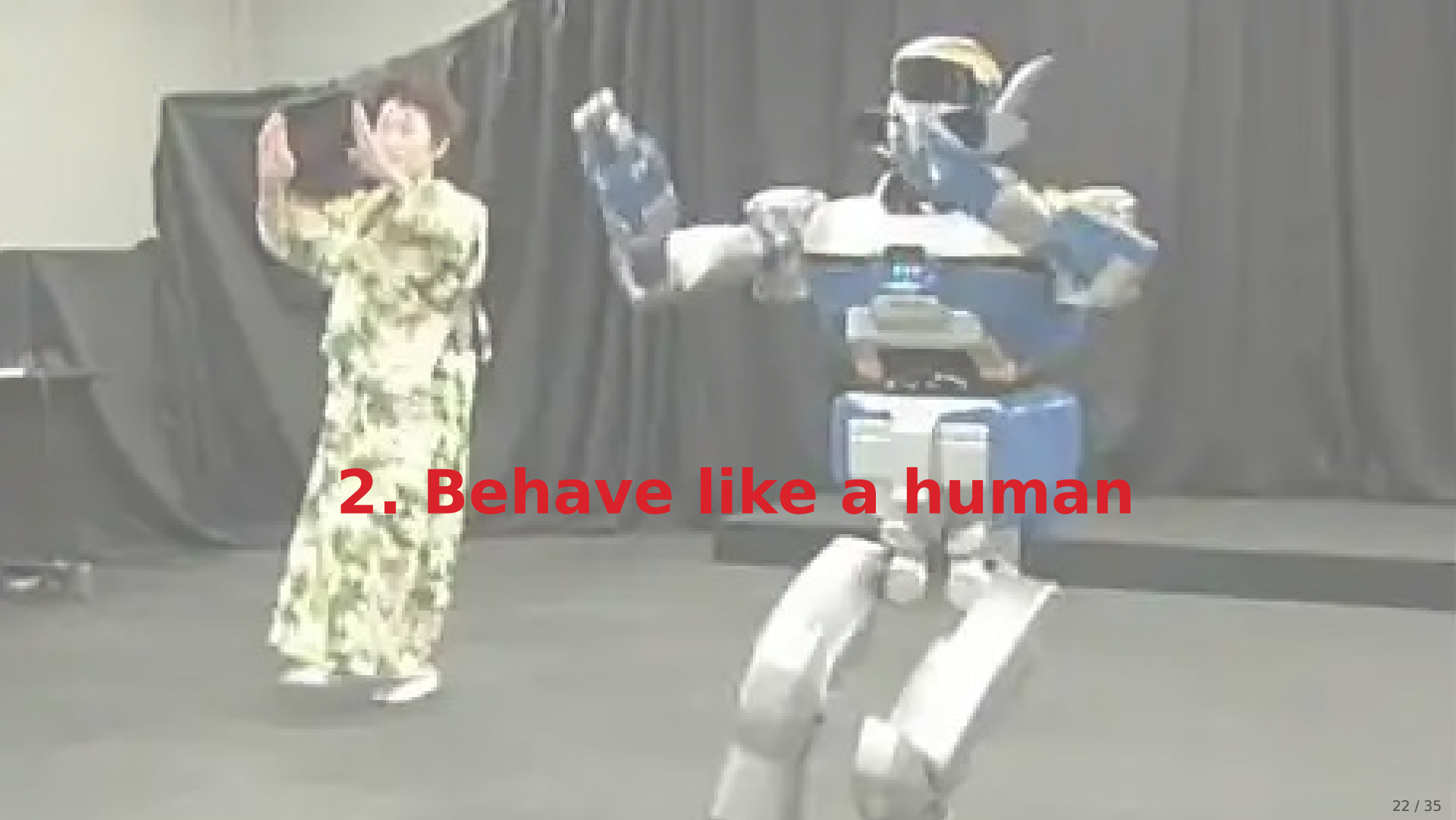
result

Validation: hide `webdriver` attribute

Table 2: Results from the screenshot evaluation.

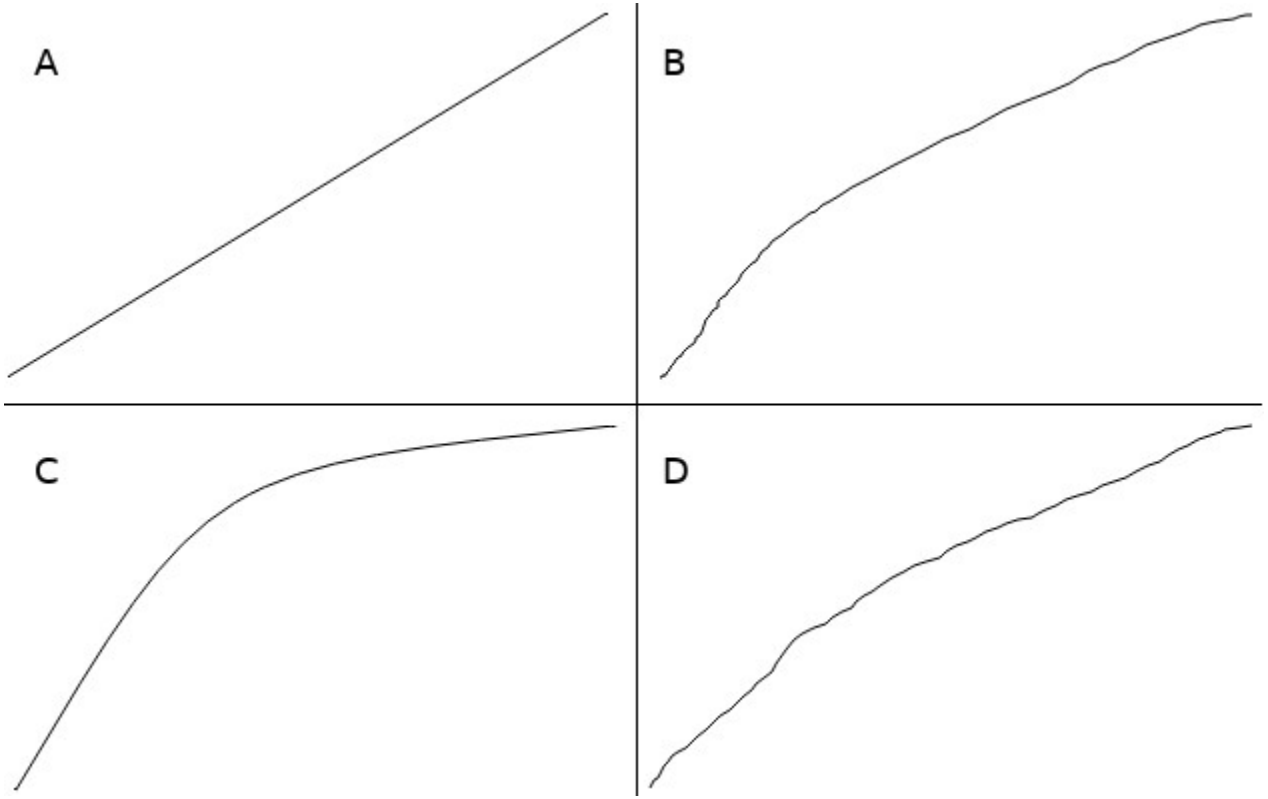
Response	sites		visits	
	(1)	(2)	(1)	(2)
total	**921**	**921**	**7,230**	**7,221**
missing ads	7	3	56	10
– no ads	5	1	40	4
– less ads	2	2	16	6
blocking/CAPTCHAs	8	1	49	3
frozen video element(s)	1	0	8	0

Results for crawler OpenWPM (1) and OpenWPM+extension (2).



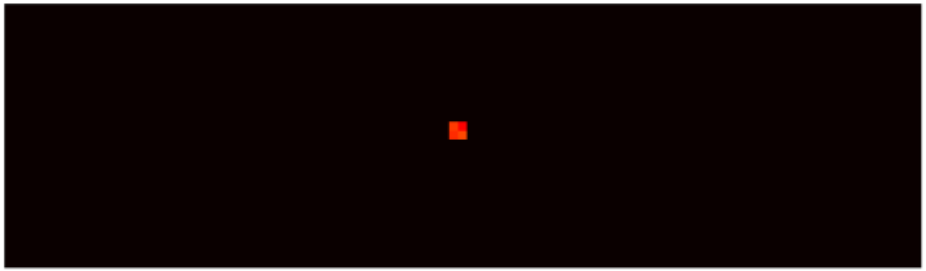
2. Behave like a human

Quiz-time! How many humans?

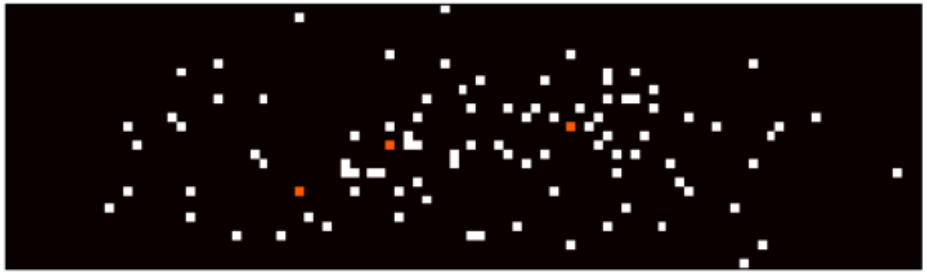


Quiz: how many humans?

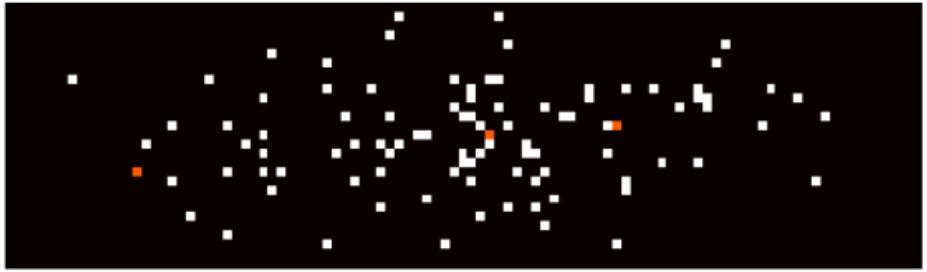
a



b



c



d

HLISA vs Selenium

- **Mouse movement**
  - **Was:** straight line, 1 speed
  - **Is:** bezier curve with jitter based on human jitter, acceleration + deceleration

HLISA vs Selenium

- **Mouse movement**
  - **Was:** straight line, 1 speed
  - **Is:** bezier curve with jitter based on human jitter, acceleration + deceleration
- **Clicking**
  - **Was:** dead center of element
  - **Is:** normal distribution for (x,y); parameters based on human behaviour

HLISA vs Selenium

- **Mouse movement**
  - **Was:** straight line, 1 speed
  - **Is:** bezier curve with jitter based on human jitter, acceleration + deceleration
- **Clicking**
  - **Was:** dead center of element
  - **Is:** normal distribution for (x,y); parameters based on human behaviour
- **Scrolling**
  - **Was:** not available
  - **Is:** scroll wheel simulation with longer break for “moving finger”

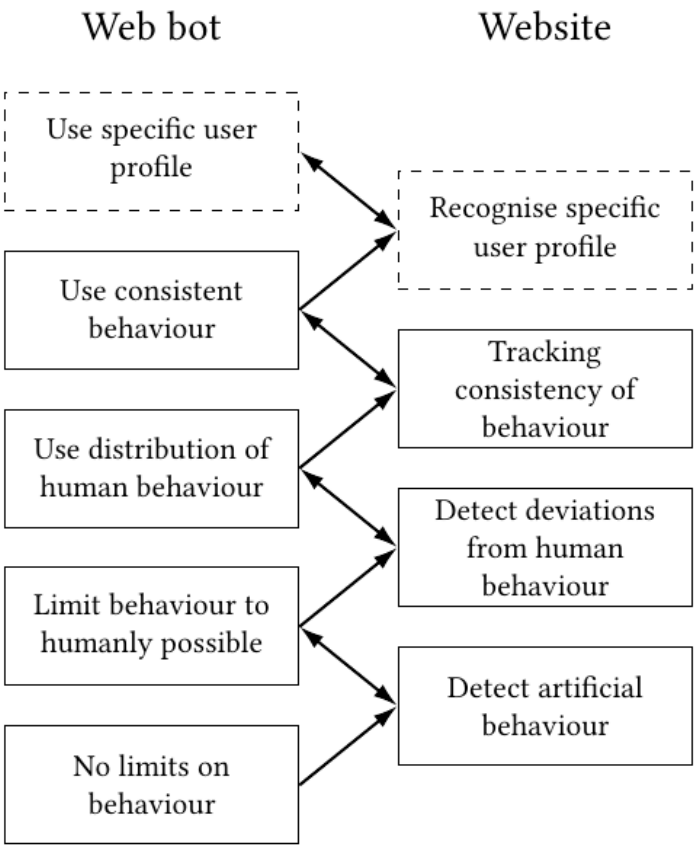
HLISA vs Selenium

- **Mouse movement**
  - **Was:** straight line, 1 speed
  - **Is:** bezier curve with jitter based on human jitter, acceleration + deceleration
- **Clicking**
  - **Was:** dead center of element
  - **Is:** normal distribution for (x,y); parameters based on human behaviour
- **Scrolling**
  - **Was:** not available
  - **Is:** scroll wheel simulation with longer break for “moving finger”
- **Typing**
  - **Was:** 13,333 char/min; no Shift key needed
  - **Is:** dwell time normally distributed, contextual pauses, Shift key used

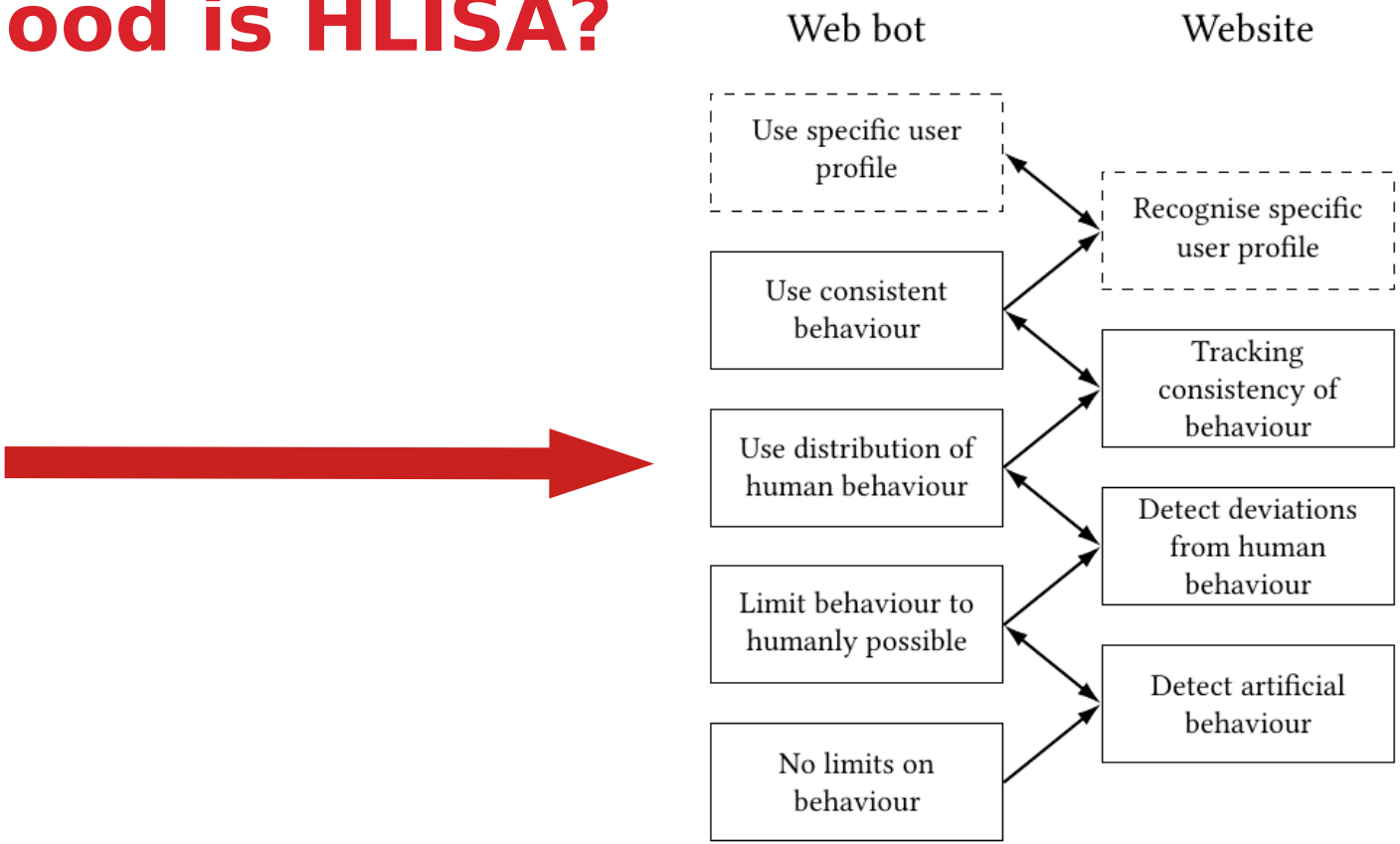
How good is HLISA?

- How much detection can HLISA withstand?
  - Or: what level of detection is required to detect HLISA?

How good is HLISA?



How good is HLISA?



Wrapping up

Ethical aspects

- **Use of human data**
  - Only one subject…
    - …reliability of measurements
- **Collateral damage potential**
  - HLISA may improve malicious bots
    - Various clickfraud bots seem to be at similar level (or better)
  - Interaction model may be used beyond intended scope
    - Stipulate not to do this

Conclusions & future work

- We’re ready to use stealth bots now!
  - Look like humans (javascript proxy objects)
  - Behave like humans
    - Typing, mouse movement, clicking, scrolling

Future work

- Measure effect: repeat study
- Arms race model suggests certain levels of detection could fall under GDPR

Questions?

HLISA: towards a more reliable measurement tool

Daniel Goßen, Hugo Jonker, Stefan Karsch, Benjamin Krumnow, David Roefs.

IMC’2021.