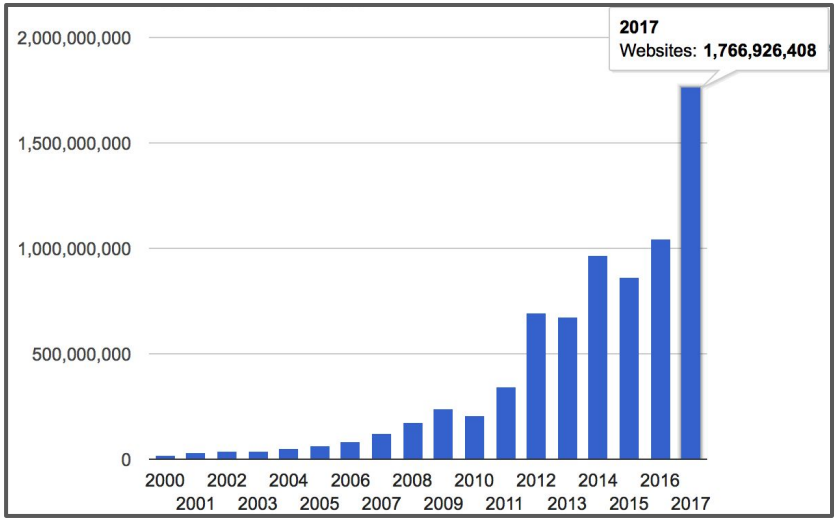
A Machine Learning Approach to Generating Security Test Inputs

Patrick Olekas, Siemens, Milford OH
Nan Niu & Wentao Wang, Univ. of Cincinnati, OH

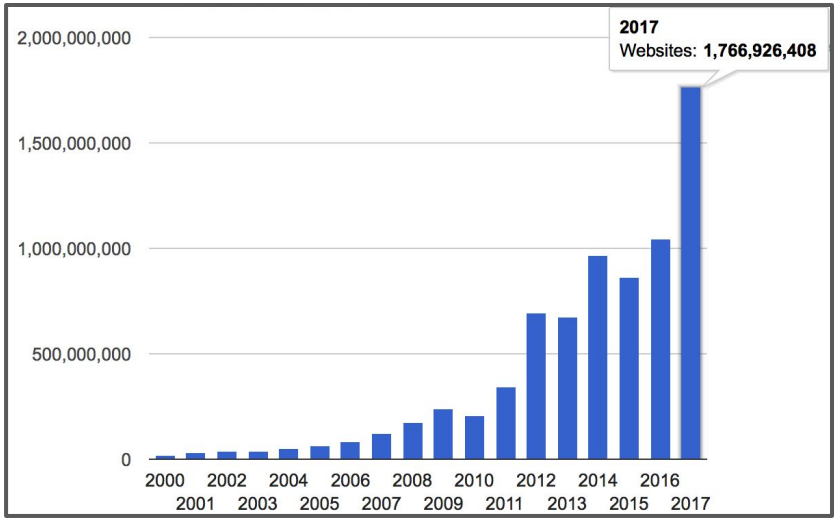
Web applications



Total number of websites
http://www.internetlivestats.com/total-number-of-websites/

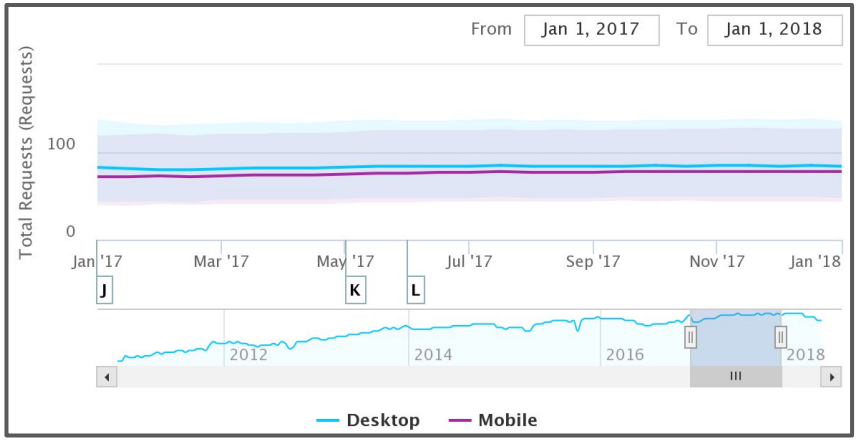
According to SiteLock Website Security Insider Q1 2018, there are 424 vulnerable pages per website (XSS)

Web applications & requests



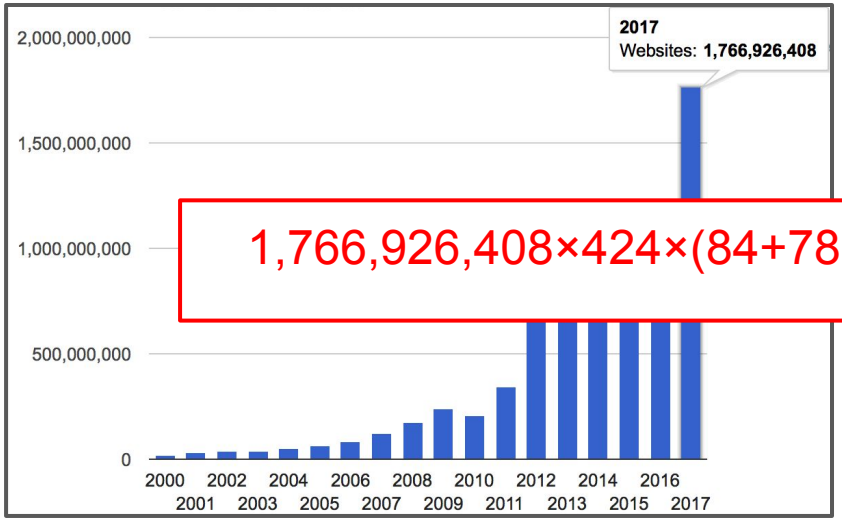
Total number of websites
http://www.internetlivestats.com/total-number-of-websites/

According to SiteLock Website Security Insider Q1 2018, there are 424 vulnerable pages per website (XSS)

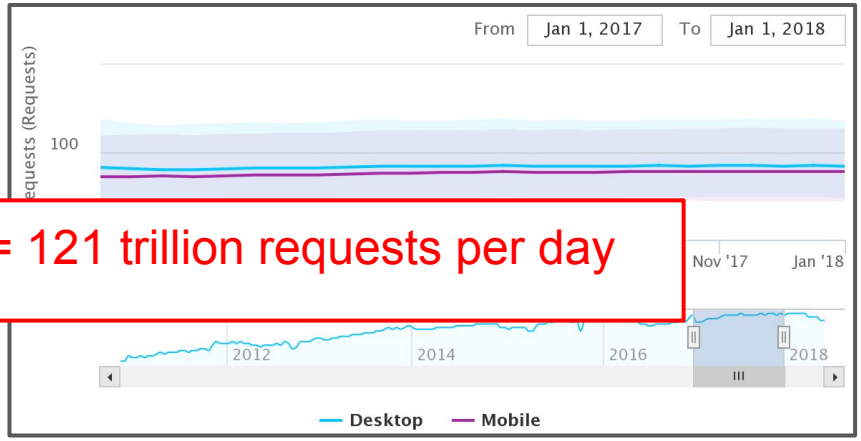


Average number of requests per page each day (Jan 1, 2017--Jan 1, 2018):
84 requests (from desktop)
78 requests (from mobile)
http://httparchive.org/reports/page-weight?start=2017_01_01&end=2018_01_01

Requests to XSS vulnerable pages



1,766,926,408×424×(84+78) = 121 trillion requests per day



Total number of websites
http://www.internetlivestats.com/total-number-of-websites/

According to SiteLock Website Security Insider Q1 2018, there are 424 vulnerable pages per website (XSS)

Average number of requests per page each day (Jan 1, 2017--Jan 1, 2018):
84 requests (from desktop)
78 requests (from mobile)
http://httparchive.org/reports/page-weight?start=2017_01_01&end=2018_01_01

eBay data breach via XSS vulnerability

Hackers were able to steal nearly 150 million accounts information.
https://www.tripwire.com/state-of-security/latest-security-news/hackers-redirected-ebay-shoppers-to-phishing-scam/

By the end of 2020, the annual cost of data breaches at the global level will skyrocket to $2.1 trillion, according to Juniper Research, a U.K.-based market analysis firm.
http://news.cuna.org/articles/105948-data-breach-costs-will-soar-to-2t-juniper

Siemens security monitoring system

Principle for IT managers:

IT managers must discover vulnerabilities quickly and then take countermeasures (e.g., executing attacks in sandbox environment)

Siemens security monitoring system:

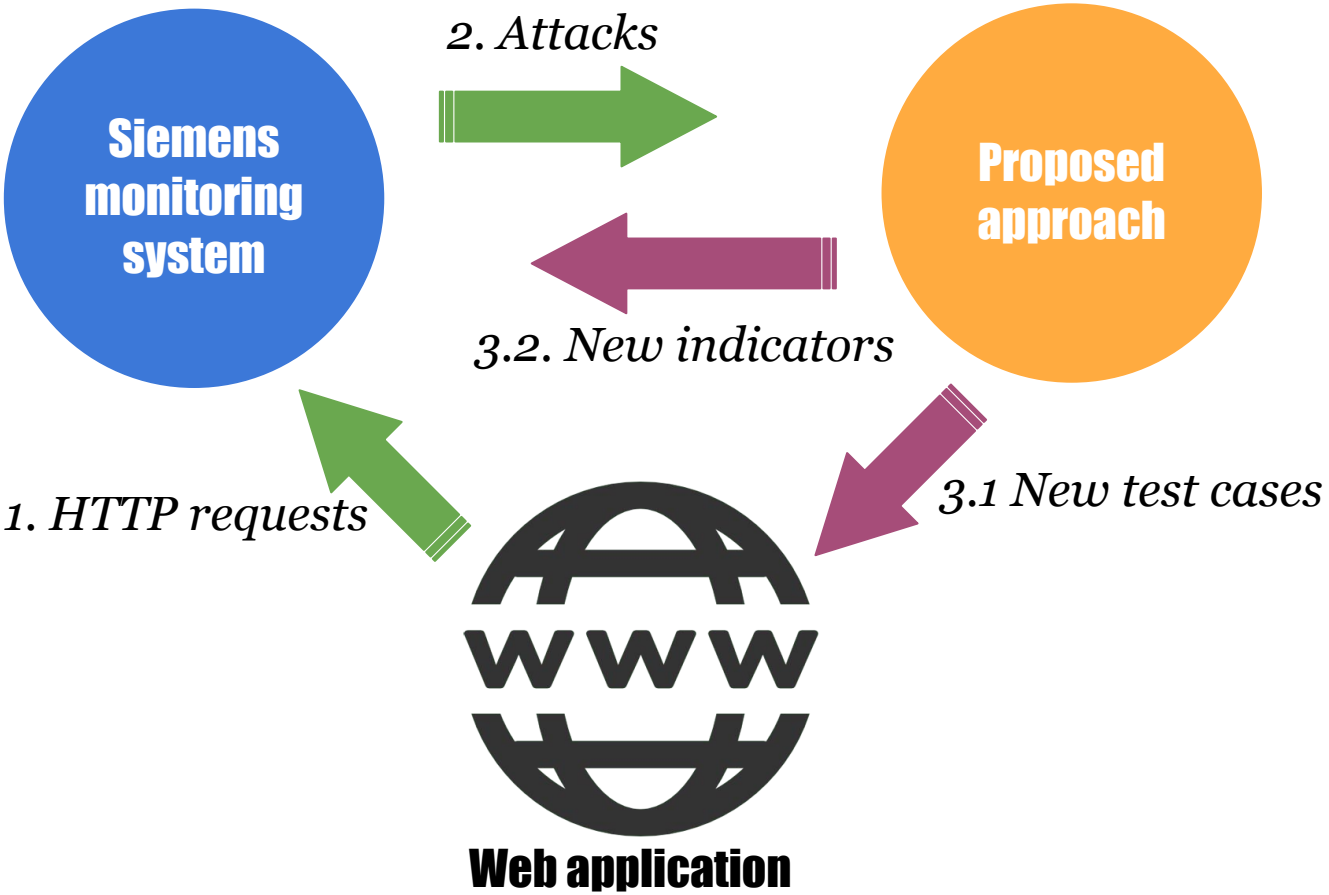
Identify cyber attacks in close to real time

Scanning data for anomalies:

- Large quantities of data moving at unusual times
- Commands that are executed countless times in succession
- Users who only work during the day according to historical data suddenly log in at night
- Unusual link redirection/leaving website

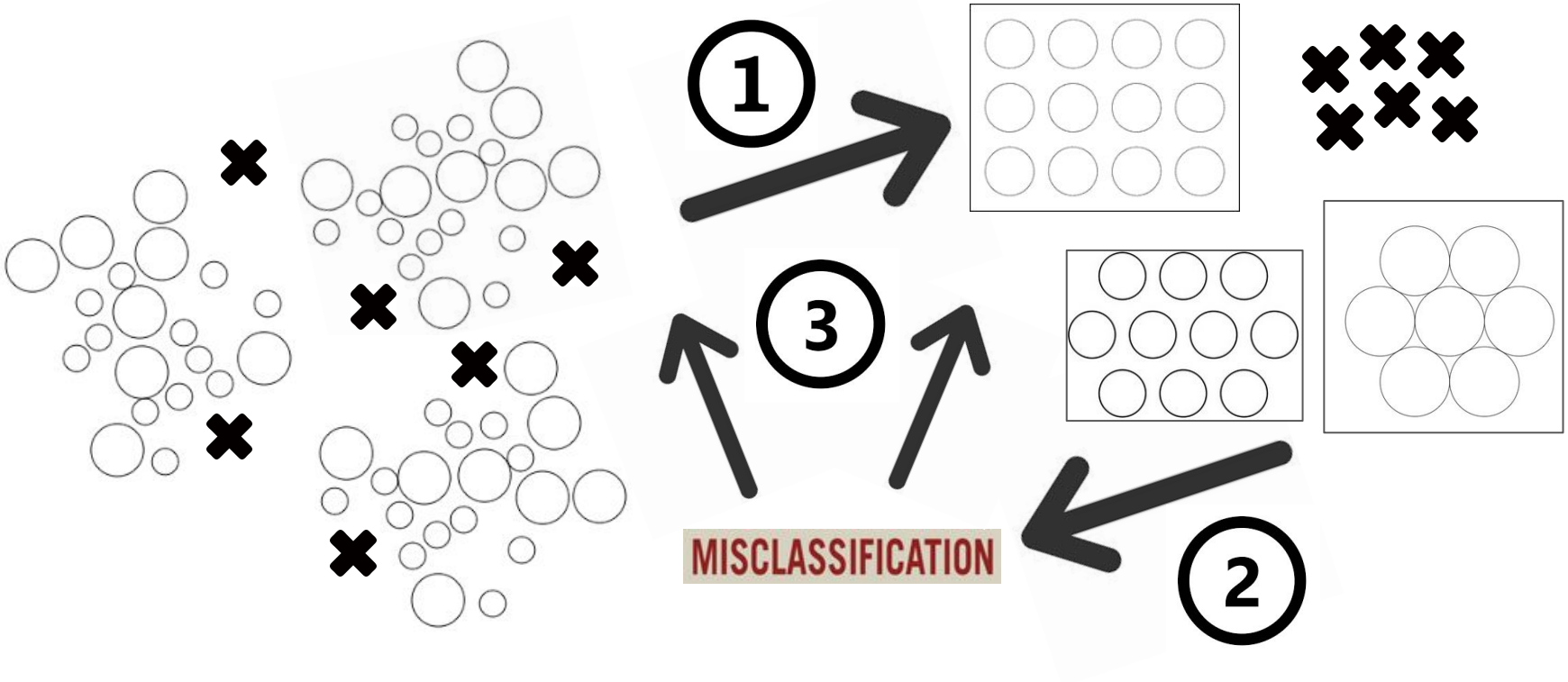
https://www.siemens.com/innovation/en/home/pictures-of-the-future/digitalization-and-software/it-security-ct-solutions.html

A synergistic scenario



Proposed approach: overview

Aim: to use known inputs (normal and malicious requests) to create new tests



Step 1: Unsupervised learning

Data characteristics: Skewed (e.g., |normal requests| / |attacks| = 1560)

Feature-based clustering

	feature1	feature2	feature3	...	feature_m
object1					
object2					
...					
object_n					

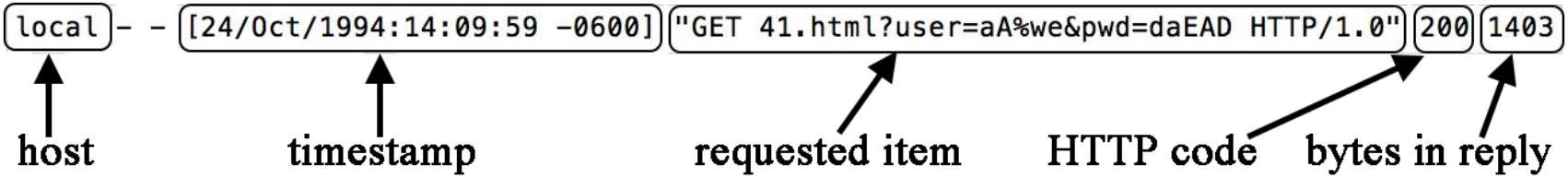
Feature examples: Request host (remote/local), request time (HH:MM:SS), request length, requested item type (e.g., html and jpg), returned item size

Calgary dataset

Description:

This dataset contains approximately one year's worth of all HTTP requests (726,739) to the University of Calgary's Department of Computer Science WWW server located in Calgary, Alberta, Canada.
http://ita.ee.lbl.gov/html/contrib/Calgary-HTTP.html

Format:



Feature-based clustering

Alg.s: hierarchical (agglomerative vs. divisive), centroid-based (*k*-means), ...

Characteristics: O(n^3) for agglomerative, O(2^(n-1)) for divisive, NP-hard, ...

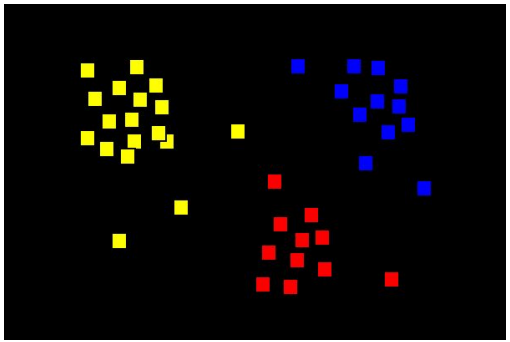
When to stop, |C|, outlier, ...

Our guiding criteria: |C_normal| is balanced and

|C_normal| = |attacks|

Clustering results

|Attacks|	|Objects|	K	|C|		
	Randomly choose: 3200		mean±s.d.	max	min
145		22	151±35	243	105



Manual labeling:

Long HTTP header parameters; image requests (e.g., gif); HTTP error code (e.g., 404), and others (mixed; unseen/unknown)

Step 2: Supervised learning

Classification: on top of the attack group and any normal-request cluster

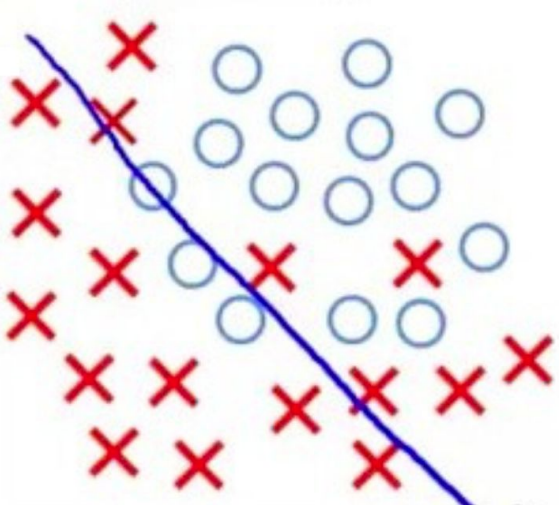
Key tenet: by using the same feature set, some form of the inverse relationship may hold

Alg.s: statistical (linear (LR) vs non-linear (SVM) vs kernel density estimation (kNN)), decision tree learning (C4.5) ...

Characteristics: $O(n)$ for LR, $O(n^2) \sim O(n^3)$ for SVM, $O(nd)$ for kNN, $O(n) \sim O(n \log n)$ for C4.5...

Our selection criteria: Both k-means and LR are feature-based, based on Euclidean space ⇒ LR may be inverse to k-means

Classification results



Classification stopping criterion:

when the sum of the absolute values of the weight

differences is less than some small number, e.g. 10^-6

Training/testing sets: N-fold (N=10)

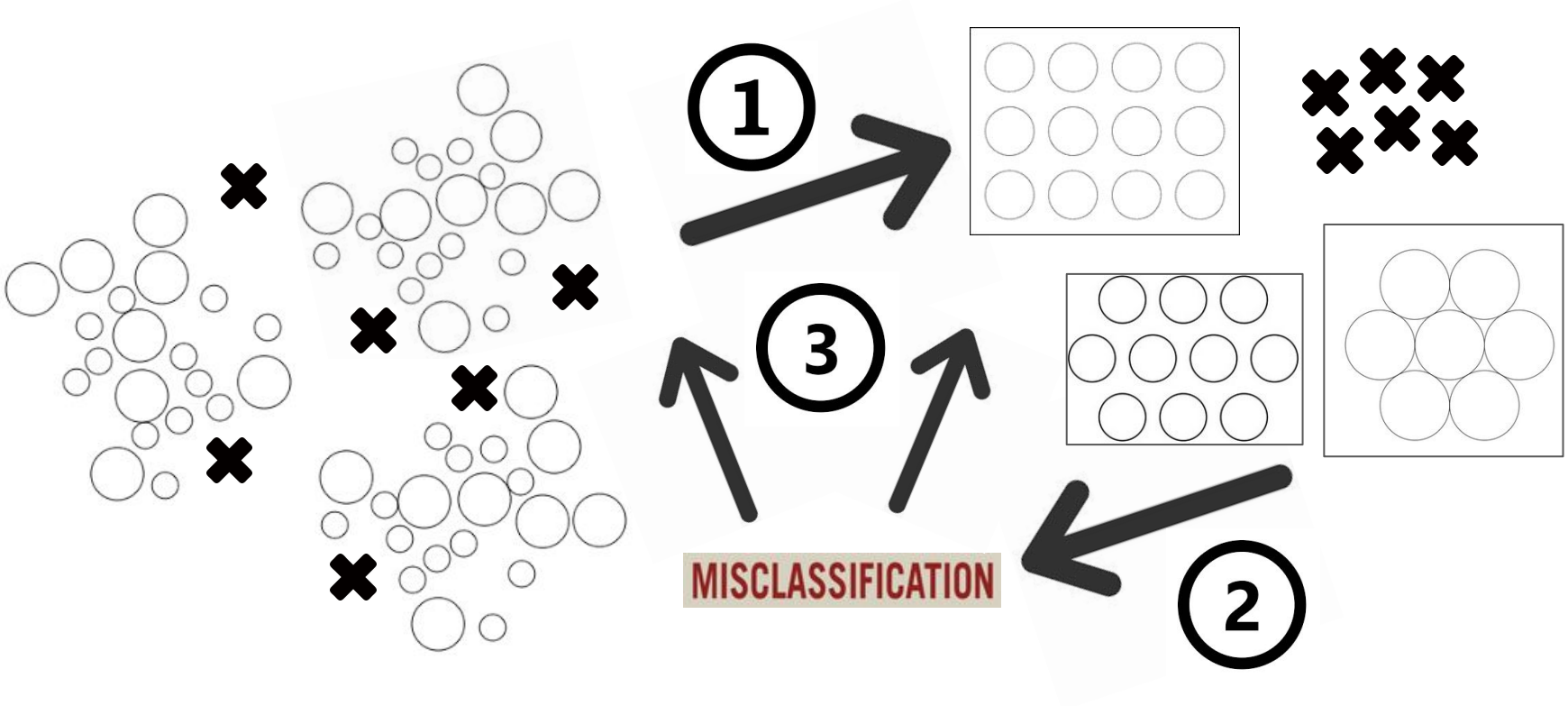
Classification performance (mean squared error rate):

mean±s.d. : 0.047±0.146; max: 0.5; min: 0.

Only 23 out of 220 classifications contain errors (i.e., all 10 rounds of validation in 2 separate clusters + 3 rounds of validation in a third cluster)

Proposed approach: overview

Aim: to use known inputs (normal and malicious requests) to **create** new tests



Creativity

Being **novel** (original and unexpected) and **appropriate** (useful and adaptive to task constraints)

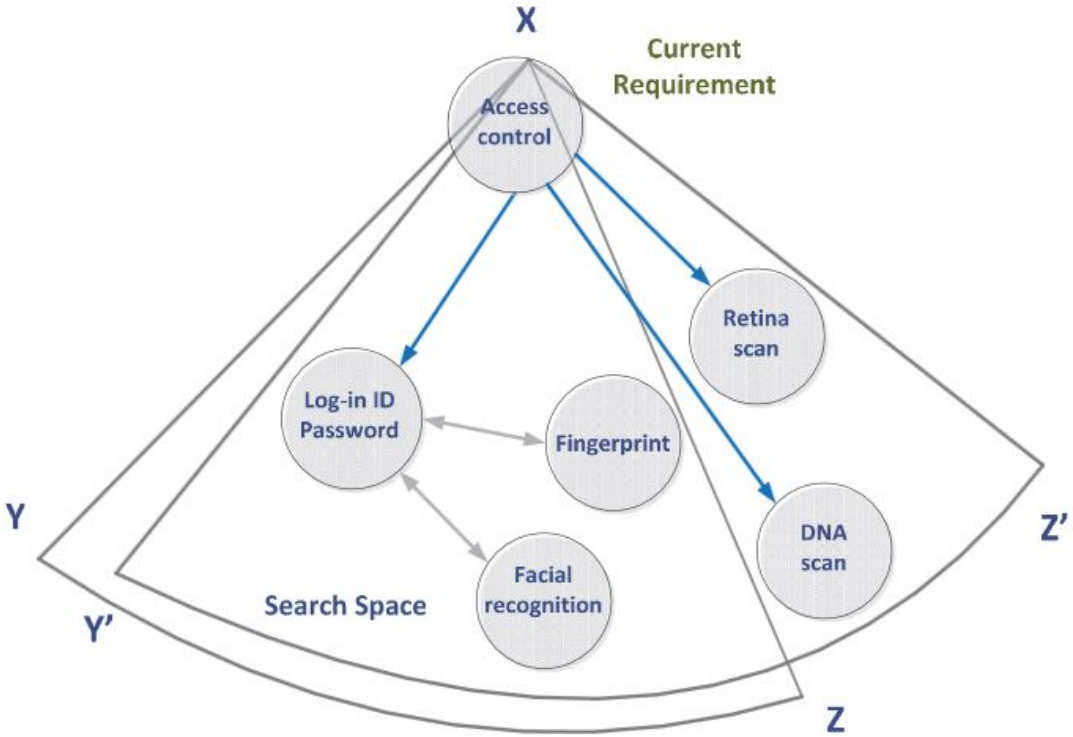
Types of creativity

H (new to a person-kind), P (new to a person), S (situated creativity)

Ways of creativity

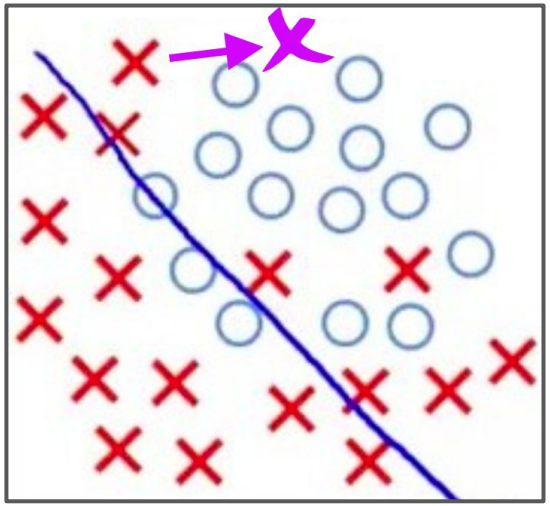
Exploratory, Combinational, Transformational

[M. A. Boden, “The creative mind: myths and mechanisms”, Routledge, 2003]



[T. Bhowmik *et al.*, “Leveraging topic modeling and part-of-speech tagging to support combinational creativity in requirements engineering”, *Req.s Eng.*, 20(3): 253-280, 2015]

Exploratory Creativity



Misclassification (e.g., an attack ⇒ normal request):

local -- [29/Oct/1994:10:11:31 -6100] “GET index.html?<svg onload=%cookie%> HTTP/1.0” 200 304

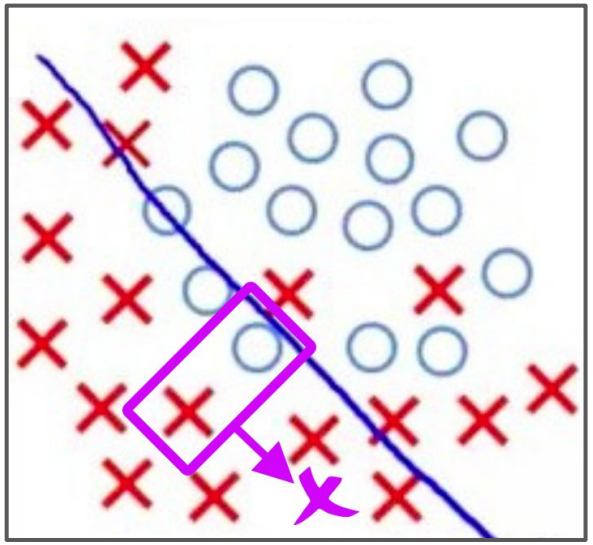
Normal-request cluster (# 5: |objects|=123):

Most normal requests and attack contains hidden information like cookie

remote -- [25/Oct/1994:16:32:01 -6100] “GET index.html HTTP/1.0 Set-Cookie: username=xxx” 200 349

Exploratory: local -- [11/Jul/2018:06:07:03] “GET 23.html?<svg onload=alert(document.domain+window.location.pathname)> HTTP/1.0” 300 0

Combinational Creativity



Misclassification (e.g., a normal request ⇒ attack):

remote - - [24/Oct/1994:15:05:03 -0600] “GET index.html?username=xxx&password=xxxx HTTP/1.0” 200 631

Nearest attack:

remote -- [24/Oct/1994: 18:10:21 -0600] “GET index.html?<script>confirm(1)</script> HTTP/1.0” 400 0

Combinational:

remote -- [24/Oct/1994: 18:10:21 -0600] “GET index.html?<script>prompt(‘Confirm password’)</script> HTTP/1.0” 400 0

Tool Support: Snuck

Description: Snuck is an open-source automated tool that can help find XSS vulnerabilities in web applications.

Source code and tutorial: https://github.com/mauro-g/snuck/wiki

Key components:

User can define testing flow in a xml file (e.g., usecase.xml)

Test case inputs are saved in payloads folder

Testing results report in a document (e.g., report.html)

More information:

F. d'Amore and M. Gentile, "Automatic and Context-Aware Cross-Site Scripting Filter Evasion", Sapienza University of Roma, Technical Report, no. 4, 2012.

Demo

Step 1: Run ‘Target T’ project in server

Step 2: Demo Target T

Step 3: Run ‘snuck->src->core->Starter.java’ as Java Application

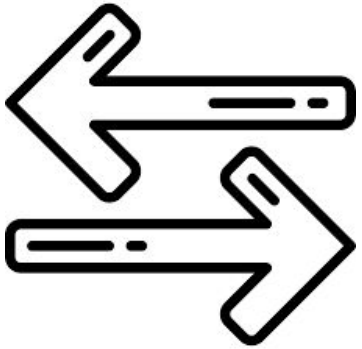
Step 4: Display ‘C:/Users/niunn/workspace/snuck/report.html’

Step 5: Copy test case in ‘payloads-new’ to the ‘payloads’ and rename report.html

Step 6: Rerun Starter.java and display report.html again

Step 7: Manually show two new test cases in web browser (i.e., <svg onload=alert(document.domain+window.location.pathname)> and <script>prompt(‘Confirm password’)</script>)

Take-away messages



novel &
appropriate