Aalto University

Fast client-side phishing detection

A case-study in applying machine learning to solve security/privacy problems

N. Asokan
(joint work with Samuel Marchal, Giovanni Armano, Kalle Saari, Tommi Gröndahl, Nidhi Singh, Mika Juuti)

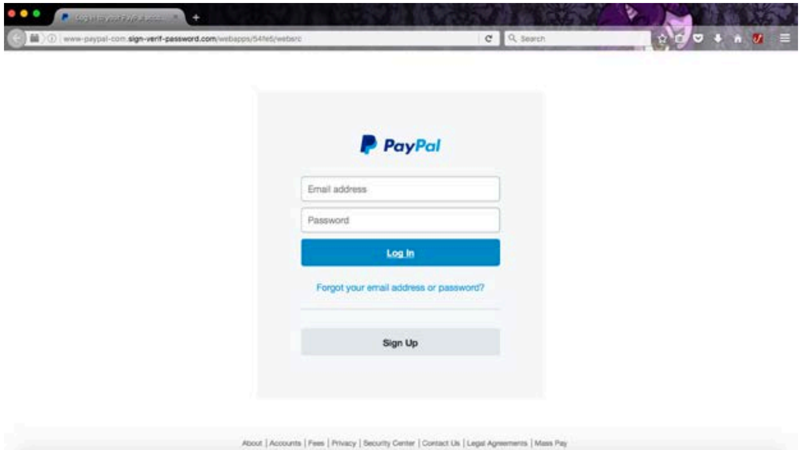
Outline

Off-the-Hook: a client-side phishing detection technique

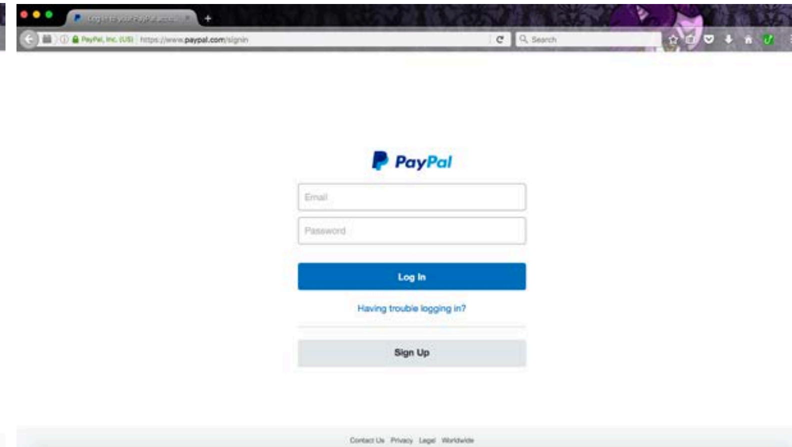
Lessons learned

- Pitfalls in applying machine learning to security/privacy problems
- Ways of avoiding pitfalls
- (From the perspective of system security experts)

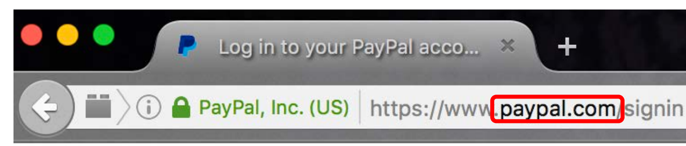
Phishing webpages



Phishing webpage (phish)



Legitimate webpage



State of the art in phishing detection

Centralized black lists

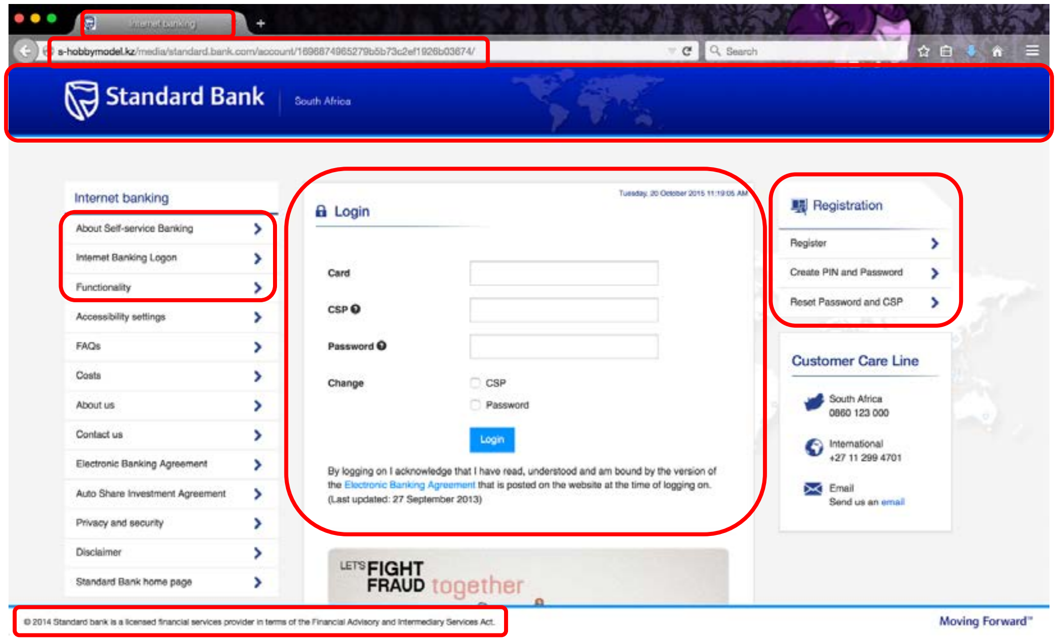
- vulnerability to "dynamic phishing": content depends on client
- Update time lag
- threat to user privacy

Google Safe Browsing McAfee SiteAdvisor WOT ...

Application of machine learning

- may not have "temporal resilience": accuracy degrading with time

Data sources on a webpage



Starting URL

Landing URL

Redirection chain

Logged links

HTML source code:

- Text
- Title
- HREF links
- Copyright

Phisher’s control & constraints

Data sources differ in terms of the levels of

- control the phisher has over a source
- constraints placed on the phisher in manipulating that source

URL Structure

$$\textit{protocol://}\overbrace{\textit{[subdomains.]}}^{\text{FreeURL}}\overbrace{\textit{mld.ps}}^{\text{Registered Domain Name}}\overbrace{\textit{[/path][?query]}}^{\text{FreeURL}}$$

https://www.amazon.co.uk/ap/signin?_encoding=UTF8

- Protocol = *https*
- Registered domain name (RDN) = *amazon.co.uk*
- Main level domain (*mld)* = *amazon*
- FreeURL = *{www, /ap/signin?_encoding=UTF8}*

Phisher's control & constraints

Control:

- **External** loaded content (logged links) and **external** HREF links are *usually* **not controlled** by page owner.

Constraints:

- **Registered domain name** part of URL cannot be freely defined: **constrained** by DNS registration policies.

Conjectures

Improve phish detection by modeling control/constraints

- generalizable, language independent, hard to circumvent

Identity target of phish by analyzing terms in data sources

- guide users where they really intended to go

Data sources: control & constraints

	Unconstrained	Constrained
Controlled	Text Title Copyright Internal *FreeURL (2)*	Internal *RDNs* (2)
Uncontrolled	External *FreeURL (2)*	External *RDNs* (2)

Feature selection

A small set (212) of features computed from data sources:

- URL features (106): e.g., # of dots in *FreeURL*
- Consistency features (101)
- Webpage content (5): e.g., # of characters in *Text*

Features not data-driven: e.g., no bag-of-words features

- Conjecture: can lead to language-independence, temporal resilience

Consistency features

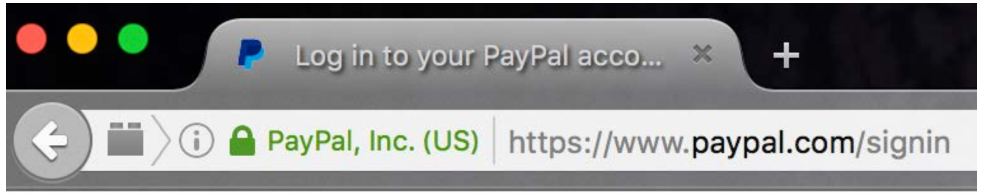
Term usage (66)

- strings of 3 or more characters, separated by standard delimiters

“Main level domain” *(mld)* usage in starting/landing URLs *(*22)

“Registered domain name” usage *(RDN)* (13)

Term usage consistency



Title: *“Log in to your PayPal account”* **RDN: *paypal.com***

$$D_{title} = \{(\text{log},0.25);(\text{your},0.25);(\text{paypal},0.25);(\text{account},0.25)\}$$

$$D_{startrdn} = \{(\text{paypal},1)\}$$

Hellinger distance

$$\text{f} = H(D_{title}, D_{startrdn}) = \frac{\sqrt{0.25 + 0.25 + (\sqrt{0.25} - \sqrt{1})^2 + 0.25}}{\sqrt{2}} = 0.71$$

Classification

Decision trees:

- Easier understanding of the decision process (intelligibility)
- Ability to learn from little training data
- Good performance with a small feature set
- No need for data normalization

Gradient Boosting (ensemble learning):

- Resilient to adversarial inference of model parameters
- Likelihood to belong to a class (score from individual learners) // no hard decision (good for tuning the decision)

→ **Fast decision**

Target identification

Identify terms representing the service/brand: keyterms

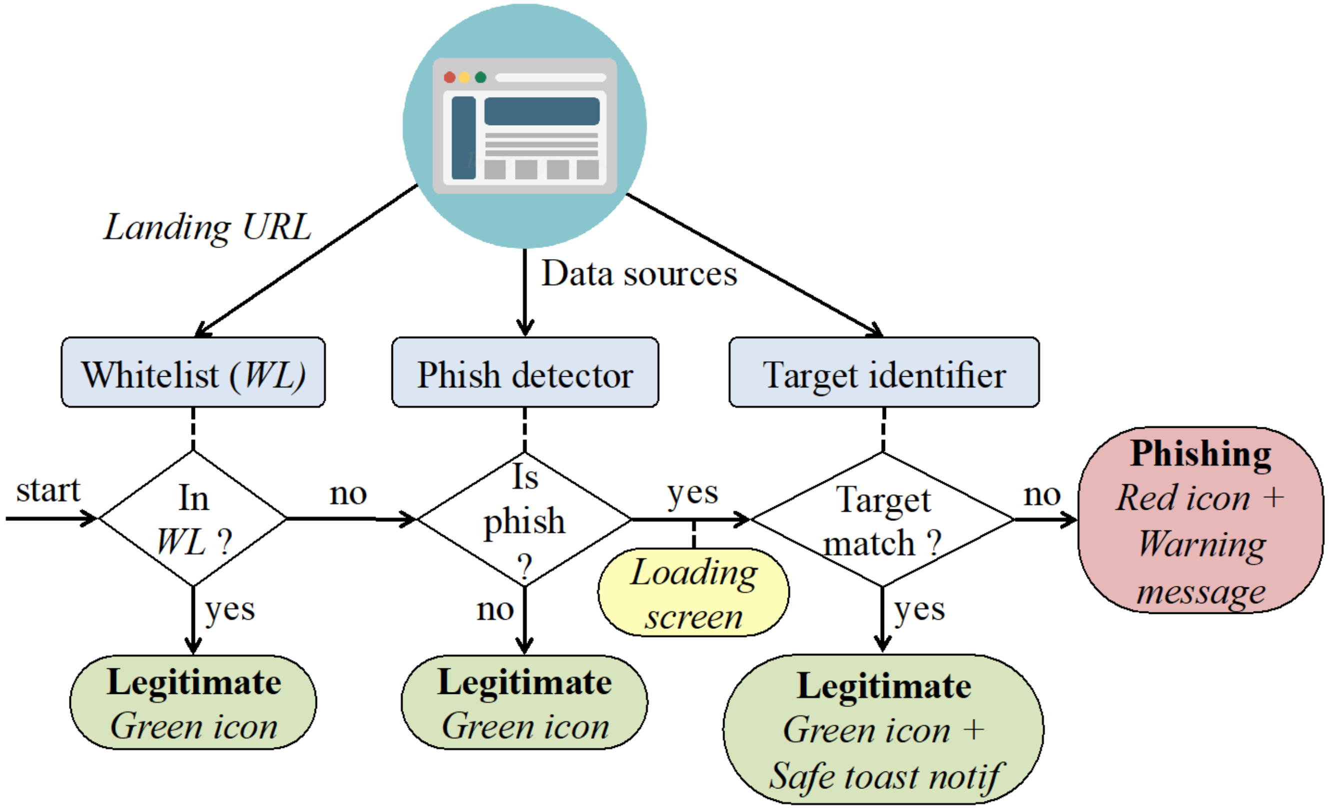
Assumption: keyterms appear in several data sources

➡ Intersect sets of terms extracted from different **visible** data sources (title, text, starting/landing URL, Copyright, HREF links)

Query search engine with top keyterms:

- Website appears in top search results → legitimate
- Else, phish; top search results ~ potential targets of phishing

Off-the-Hook anti-phishing system



Off-the-Hook browser add-on

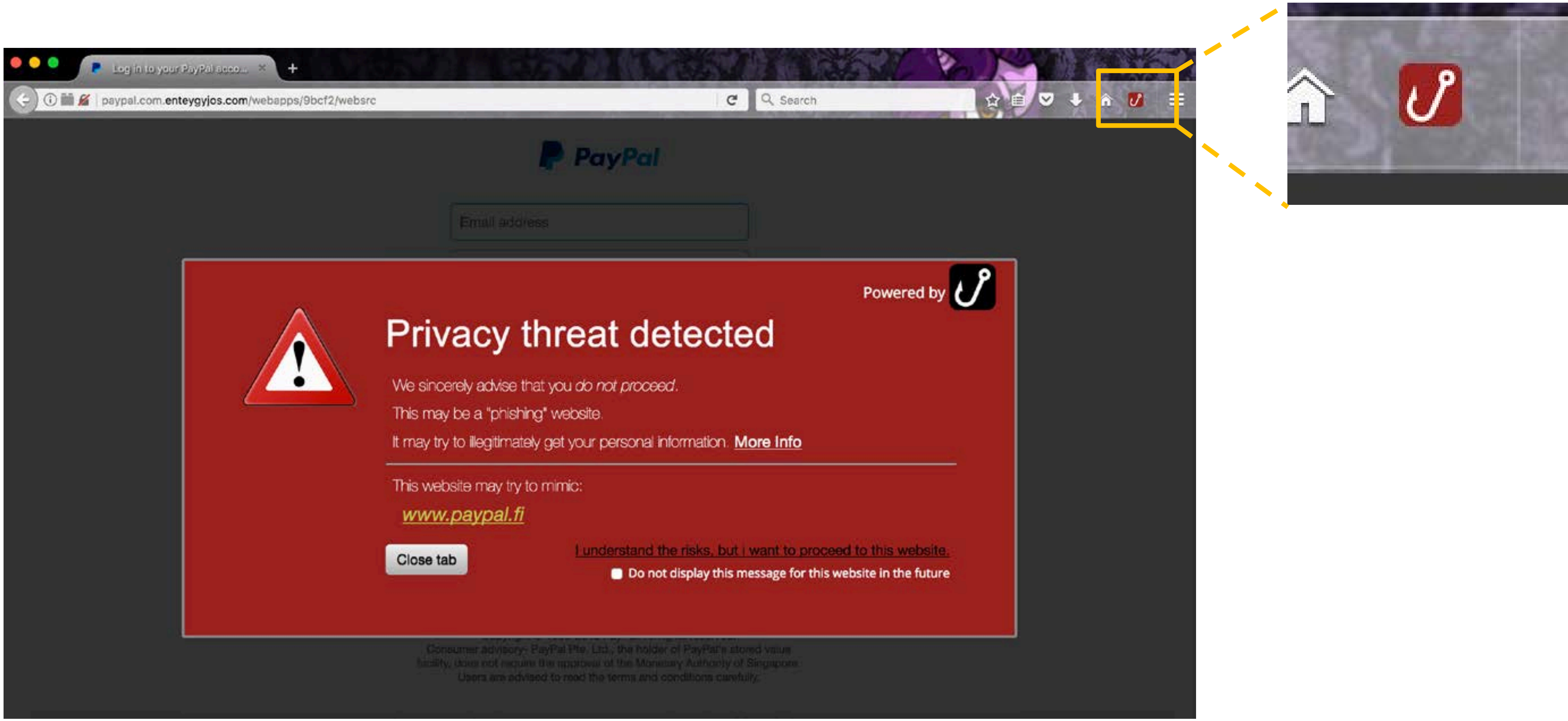
Client-side implementation

- Preserves user privacy
- Resists dynamic phishing

Multi-browser / Cross platform

- Chrome*, Firefox
- Windows (>= 8), Mac OSX (>= 10.8), Ubuntu (>= 12.04)

Off-the-Hook warning



Evaluation

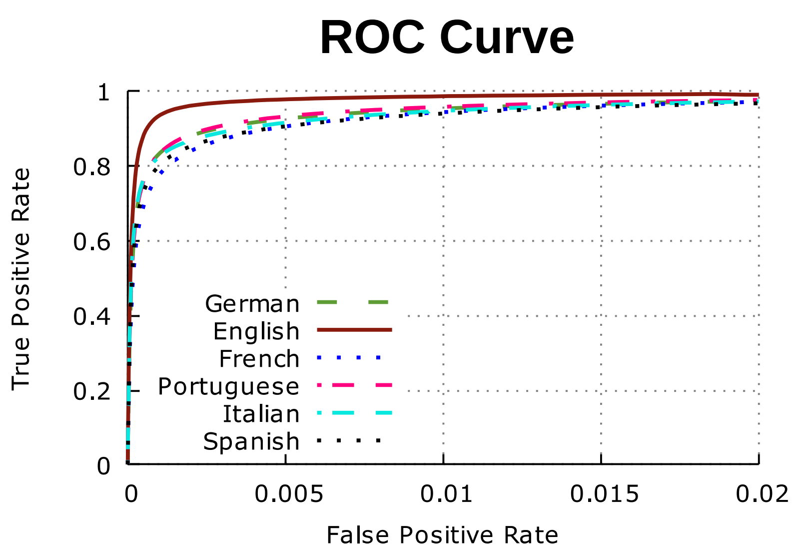
Classifier Training:

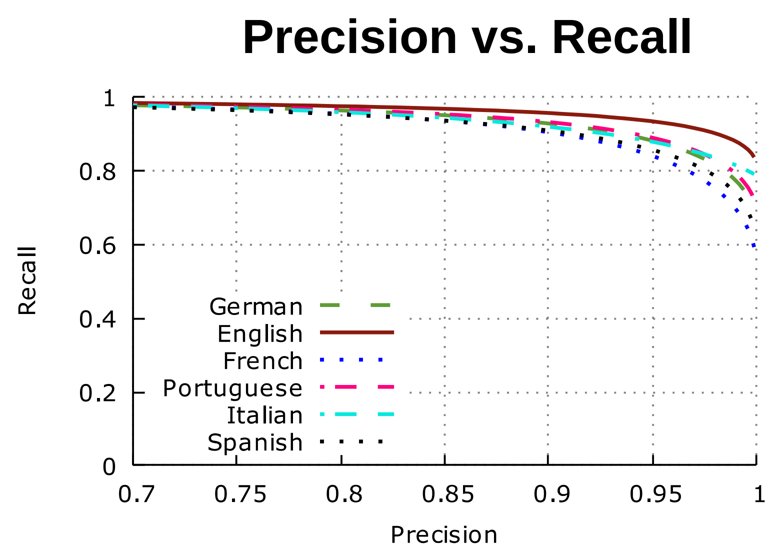
- 8,500 legitimate webpages (English)
- 1,500 phishing webpages (taken from PhishTank & manually verified)

Evaluation:

- Legitimate webpages:
  - 100,000 English
  - 20,000 each in French, German, Italian, Portuguese and Spanish
- 2,000 phishing webpages (PhishTank; manually verified)

Classification accuracy

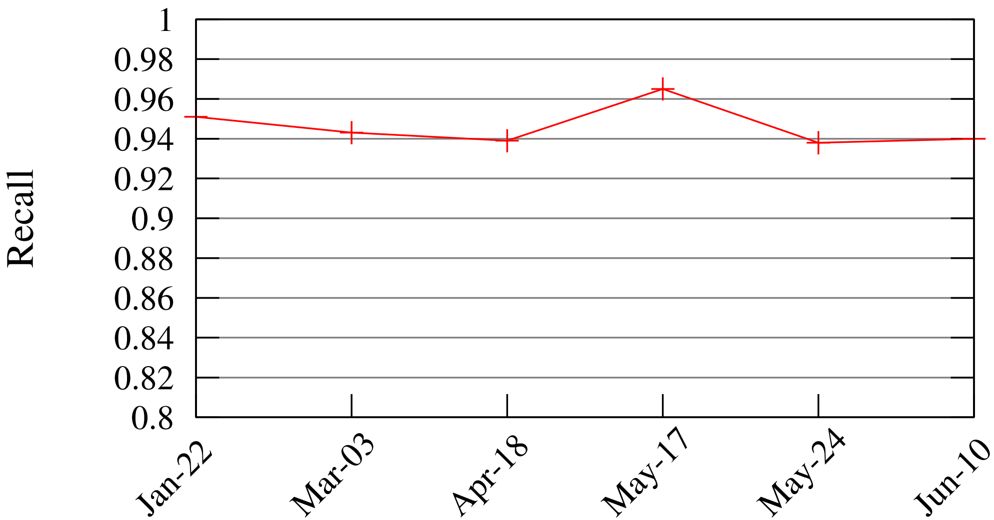




200,000 multi-lingual legit / 2,000 phishs (**≈ real world distribution**)

Precision	***Recall***	***FP Rate***	***AUC***	***Accuracy***
0.975	0.951	**0.0008**	0.999	**0.999**

Classification accuracy over time



Model trained:

- September 2015

Applied on phishs:

- January – June 2016
- ~2500 fresh, verified phishtank entries

Performance

Small memory footprint: 295 MB

Minimal impact on web surfing

- Phishing webpages:
  - Interaction blocked in **< 0.2 second**
  - Warning displayed (and target identified) in **< 2 seconds**
- Legitimate webpages:
  - **No perceptible impact** (albeit false positives)

Comparison: effectiveness

	FPR	Precision	Recall	Accuracy
Cantina (CMU)	**0.03**	**0.212**	0.89	0.969
Cantina+ (CMU)	**0.013**	0.964	0.955	0.97
Ma et al. (UCSD)	0.001	**0.998**	0.924	0.955
Whittaker et al. (Google)	**0.0001**	0.989	0.915	**0.999**
Monarch (UCB)	0.003	0.961	**0.734**	**0.866**
Off-the-Hook	**0.0008**	0.975	**0.951**	**0.999**

Comparison: dataset sizes

	Training	Testing
Cantina (CMU)	-	2,119
Cantina+ (CMU)	2062	884
Ma et al. (UCSD)	17,750	17,750
Whittaker et al. (Google)	9,388,395	1,516,076
Monarch (UCB)	750,000	250,000
Off-the-Hook	10,000	202,000

Off-the-Hook summary

https://ssg.aalto.fi/projects/phishing/

Off-the-Hook phishing website detection system:

- Exhibits language independence
- Resists dynamic phishing
- Fast: < 0.5 second per webpage (average for all webpages)
- Accurate: > 99.9% accuracy with < 0.1% false positives

Target identification system:

- Fast: < 2 seconds per webpage
- Success rate: > 90% (1 target); 97.3% (set of three potential targets)

[MSSA16] Know Your Phish: Novel Techniques for Detecting Phishing Sites and their Targets, ICDCS 2016
[AMA16] Real-Time Client-Side Phishing Prevention Add-On, ICDCS 2016
[MAGSSA17] Off-the-Hook: An Efficient and Usable Client-Side Phishing Prevention Application, (to appear) IEEE Trans. Comput., 2017

Pitfalls in using ML for security

Aalto University

Adversaries will circumvent detection

The ML model is intended to detect/counter attacks

Adversary *will* attempt to circumvent detection:

- poison learning process
- infer detection model
- mislead classifier

In Off-the-Hook:

- Modeling constraints and controls while training
- Adversary can control External RDNs!

➡ **Resistance to adversaries**

Classification landscapes are dynamic

Attacks evolve fast

Prediction instances likely differ from training instances

- E.g., Android malware evolves due to for changes in API

In Off-the-Hook:

- Avoidance of data-driven features
- Models that allow inexpensive retraining

→ **Temporal resilience**

Maintaining labels is expensive

More training data is good; but unbalanced classes typical

Data about malicious behavior difficult to obtain

- Labeling is cumbersome, requires expertise, may be inaccurate or may evolve (e.g. phishing URLs)

In Off-the-Hook:

- Manage with small training sets
- Minimize ratio of training set size to test size

➡ **Minimal training data**

Privacy concerns are multilateral

Data used for ML may be sensitive

- Sensitive information about users in
  - training data → model inversion, membership inference
  - prediction process → user profiling, e.g., in a cloud setting (ML-as-a-service)

In Off-the-Hook:

- Client-side classifier to avoid disclosure of URLs
- But model stealing may be a concern

→ **Multilateral privacy guarantees**

Predictions need to be intelligible

Ability of humans to understand why a prediction occurs

- Detection as malicious → forensic analysis
- Explain predictions to users, e.g. why access is prevented
- “Explainability” obligations under privacy regulations like GDPR

In Off-the-Hook:

- Small set of “meaningful” features
- Use of (ensemble of) shallow decision trees

➡ **Transparent decision process**

ML failures can harm user experience

Security is usually a secondary goal

Use of ML must not negatively impact usability

- Decision process should be efficient
- Wrong predictions may have a significant usability cost

In Off-the-Hook:

- Prediction effectiveness and speed
- In phishing detection, one false positive may be one too much!

➡ **Lightweight and accurate**

Skip to conclusions

Security/privacy applications: desiderata

Circumvention resistance

- Resistance to adversaries

Temporal resilience

- Resilience in dynamic environments

Minimality

- Use of minimal training data

Privacy

- Model privacy, training set privacy, and input/output privacy

Intelligibility

- Transparent decision process

Effectiveness

- Lightweight, accurate models

Skip to conclusions

Skip to PETS

On avoiding pitfalls

Model complexity

Complex, non-linear models can resist circumvention better

- Model inversion/stealing is
  - easier with linear regression, decision tree, shallow NN
  - harder with ensemble methods, deep NN
- But complex models tend to have poor
  - intelligibility
  - temporal resilience (retraining training time/data: e.g, kernel SVM, deep NN)

Apply Occam's Razor

- opt for the simplest model possible

Skip to conclusions

Skip to PETS

Model secrecy

Keeping model secret can help resist circumvention

- E.g., ML-as-a-service hides model from adversaries
- But naïve designs degrade input/output privacy of users

Adapt ML analogue of Kerchoff's desideratum?

- Keep (only) model parameters secret
- Disclose only the ML algorithm

Skip to conclusions

Skip to PETS

Feature selection

Carefully hand-crafted features can resist circumvention better

- But needs domain expertise and human input
- Automated selection: "effectiveness" not resistance to manipulation

Also can improve intelligibility and **temporal resilience**

Avoid data-driven feature selection (e.g., bag-of-words)

Skip to conclusions

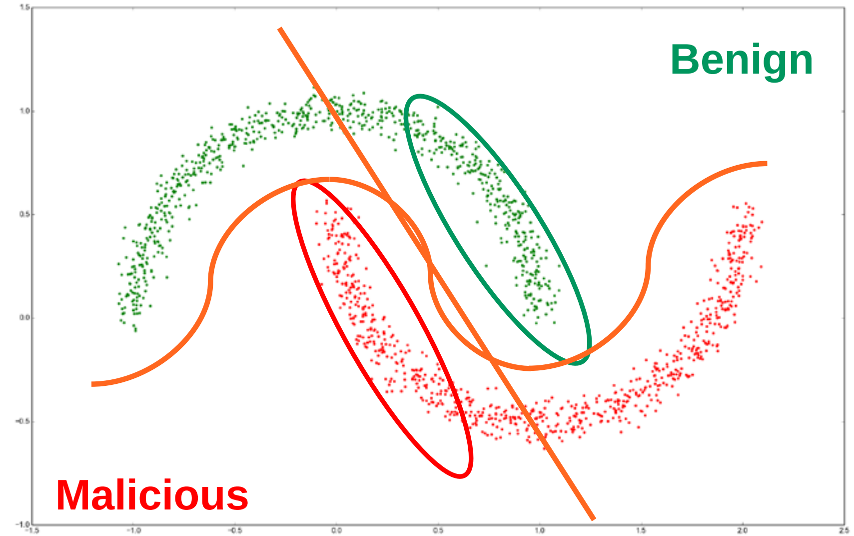
Skip to PETS

Dataset selection

Selective sampling can harm temporal resilience

- Common mistake: lack of coverage in datasets, e.g.,
  - Top 100 000 Alexa websites
  - 10,000 most popular apps + Malware that contacts malicious domains

Use representative datasets



Skip to conclusions

Skip to PETS

Evaluation approaches: datasets

Evaluation should mimic real-world usage

- Excellent academic results reportedly often fail in deployment

Use temporal separation: e.g., train on old data, test on new data

- Avoid cross-validation → can overestimate performance

Account for unbalanced class distribution

- E.g., Resampling during training, realistic distribution for testing

Skip to conclusions

Privacy-enhancing technologies

Training set privacy

- Adversary during training → training with encrypted data
- Generic membership inference attacks → differential privacy

Model privacy

- Model extraction → complex models, diff. privacy, rate limiting

Input/output privacy for predictions

- Local models (but compromise model privacy)
- MLaaS : Hide inputs/outputs from server; model from client
  - Trusted execution environments on servers (Intel SGX or other commercial TEEs)
  - Oblivious ML predictions

Skip to conclusions

Recommendations and good practice

Model selection

- Keep model secret & simple

Feature selection

- Opt for handcrafted vs. data-driven

Dataset selection

- Use representative datasets

Evaluation approaches

- Prefer temporal vs. cross-validation, use relevant metrics

Privacy-enhancing technologies

- Use local predictions, oblivious ML models, differential privacy

What about Deep Learning?

Complex decision process

- Difficult to explain decisions (intelligibility)
- Difficult to reverse engineer (circumvention resistance)

Training is complex/expensive

- Requires large amount of training data (minimality)
- Relearning is costly (temporal resilience)

Automated "feature selection"

- Adversary can impact prediction by manipulating input (circumvention resistance)

Summary

Off-the-Hook for effective phishing detection

Desiderata for using ML for security/privacy applications

Some thoughts on avoiding potential pitfalls

A little provocation!

https://ssg.aalto.fi/projects/phishing/

Additional slides

Feature selection

Rely on few features:

- Limited availability of training data (for some class at least)
- Good practice to generalize a phenomenon: 10x to 100x more training instances than features

Feature minimality

Smaller set of features ensure minimality of model

- Recall: labeled training data is difficult to obtain/maintain
- Also helps intelligibility but can ease circumvention
- Good practice dictates 10x to 100x training instances
- Size of feature set and training set depend on complexity of phenomenon being modeled

Apply Occam's Razor

- opt for the smallest feature set possible

Skip to PETS

Evaluation – dataset usage

Deal with unbalanced class problem for training

- Resample the class: under-sampling over-represented class
- Generate synthetic example for the under-represented class (e.g. SMOTE)
- Use penalized models (e.g. penalized-SVM)

Represent real-world distribution for testing

- Anomalies << normal instances (e.g. phishs << legitimate websites)
- Preserve repartition for relevant accuracy results from evaluation

Evaluation – metrics

Unbalanced class distribution impacts selection of metrics

- Accuracy, AUC, TP Rate, etc. can be high even for ineffective models

Example combination of metrics:

- Recall (TP_{rate}) → detection capability:
- Precision → reliability / usability:

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$