facebook

# Adversarial ML in Real Life

## Examples, Lessons, and Challenges

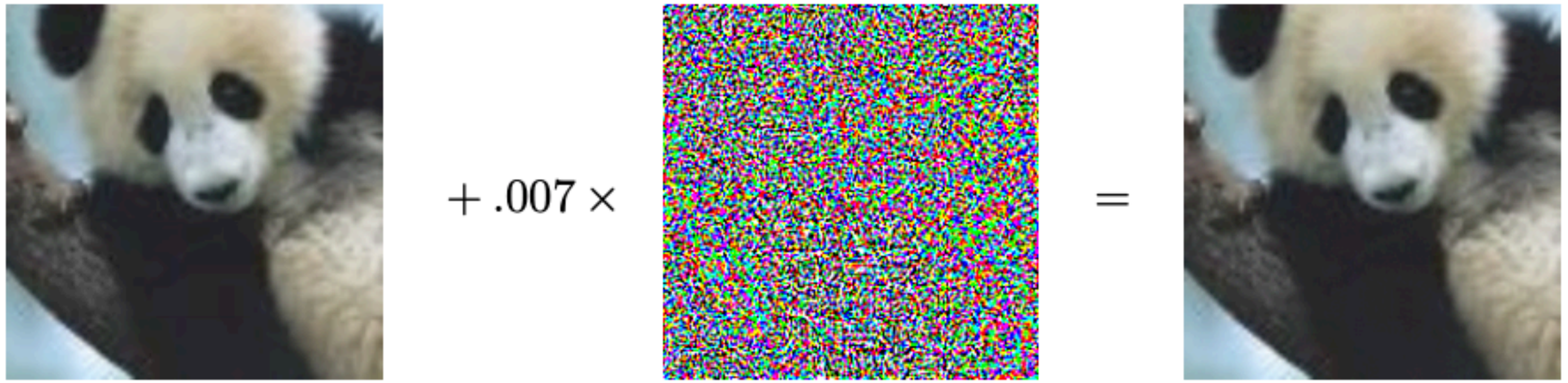
**David Freeman**

Research Scientist/Engineer, Facebook

AISecAI Workshop

London, United Kingdom, 20 January 2020

# Adversarial ML in Academia




$\boldsymbol{x}$

"panda"
57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy,

"Explaining and Harvesting Adversarial Examples," ICLR 2015

# Adversarial ML in Academia


Speed
Limit 45

Added
Lane


Speed
Limit 45


Speed
Limit 45

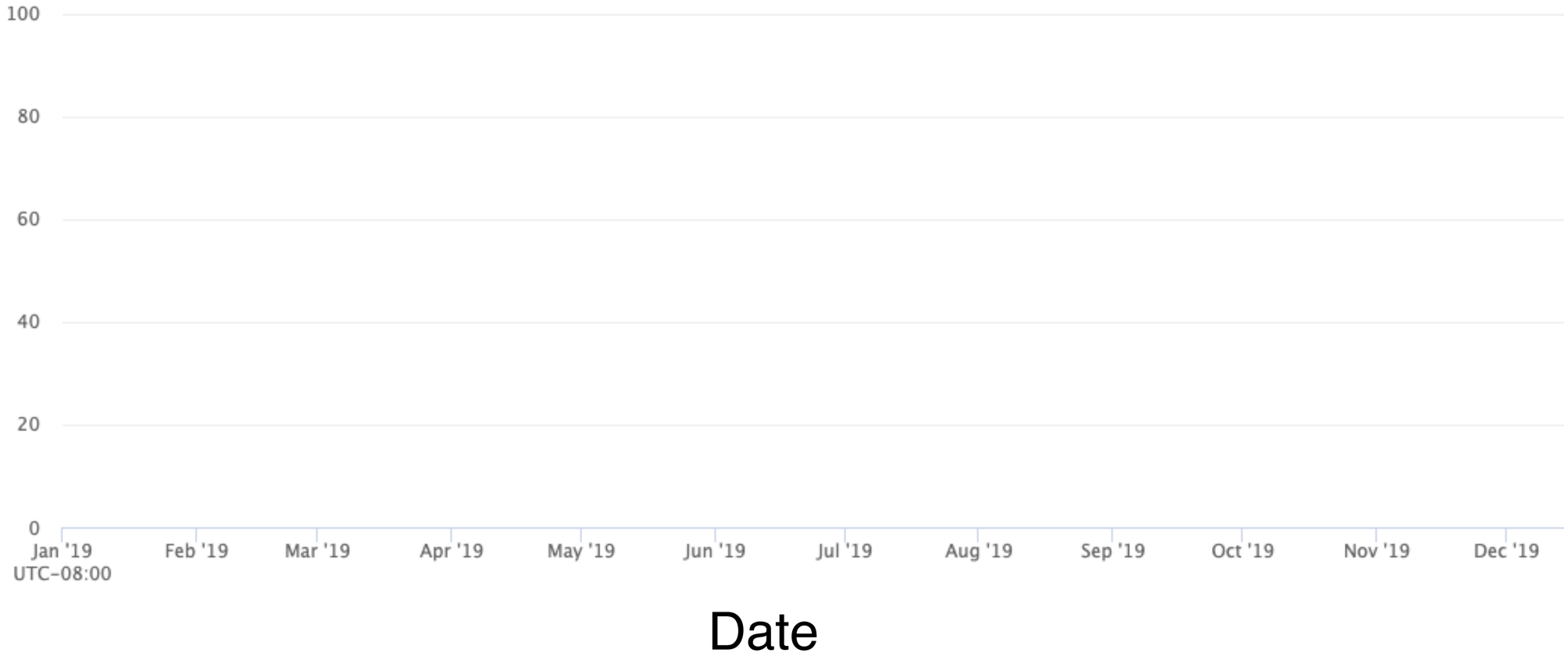

Lane
Ends

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, Dawn Song,

"Robust Physical-World Attacks on Deep Learning Visual Classification,"
CVPR 2018

# Adversarial ML on Facebook

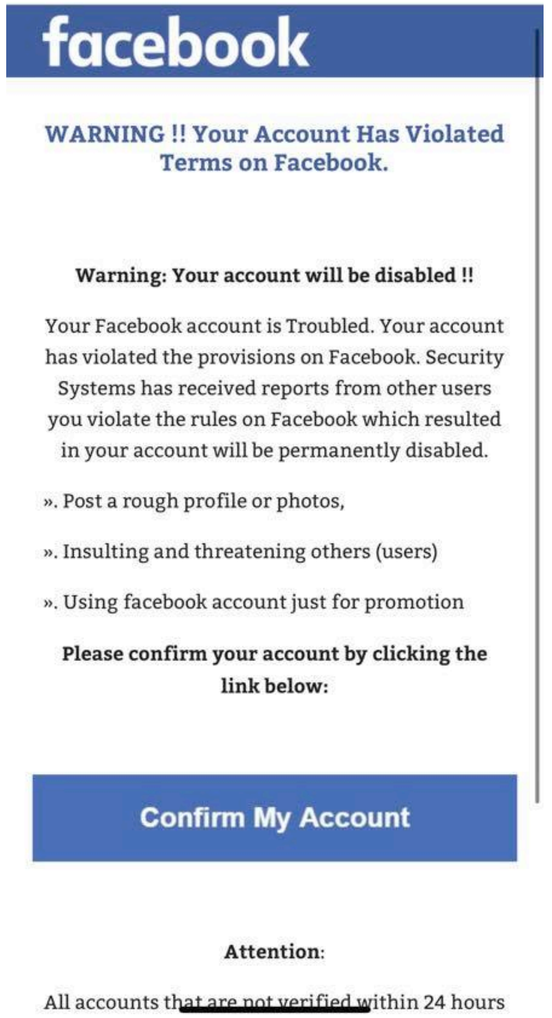
Pandas turned into Gibbons



# Adversarial ML on Facebook


Not Porn


Not Spam




Not Phishing

# Adversarial ML is Everywhere


To us, "Adversarial ML" == "ML in an Adversarial Environment"

# Fundamental Observation

# Fundamental Observation

- ML algorithms assume a stationary data distribution.

# Fundamental Observation

- ML algorithms assume a stationary data distribution.
- In an adversarial environment, deploying a model *changes the data distribution*.

# Fundamental Observation

- ML algorithms assume a stationary data distribution.

- In an adversarial environment, deploying a model *changes the data distribution.*

*All ML is broken....or is it?*

# Example I:
# A/B Testing?

# The Perils of A/B Testing

# The Perils of A/B Testing

- Fundamental A/B testing assumption:

  *Experiment effects are independent of the cohorts chosen*

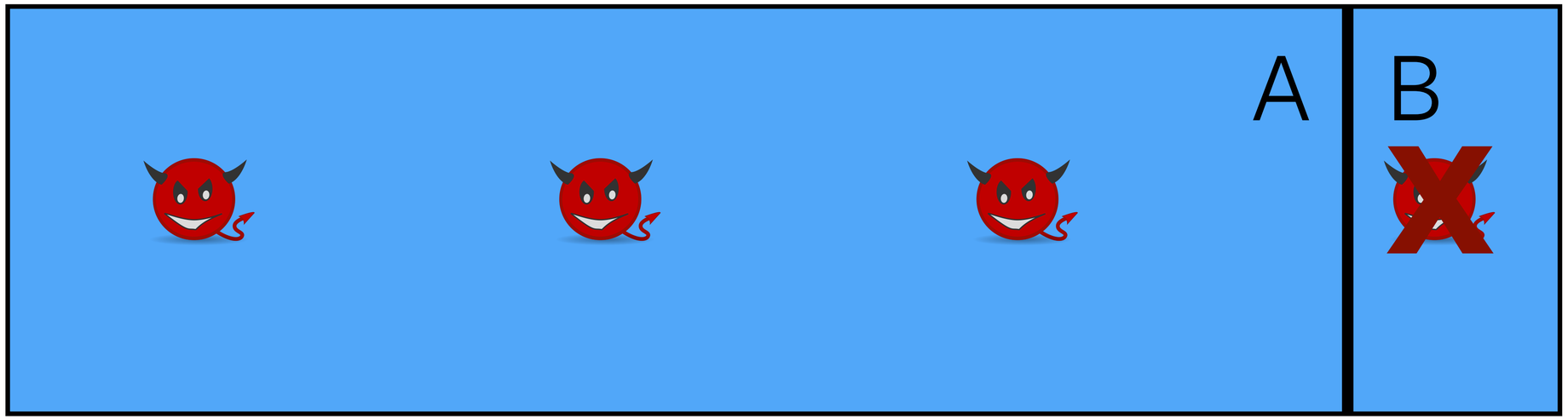
# The Perils of A/B Testing

- Fundamental A/B te[  ]assumption:

  *Experiment effects are i[ ]dent of the cohorts chosen*

# The Perils of A/B Testing

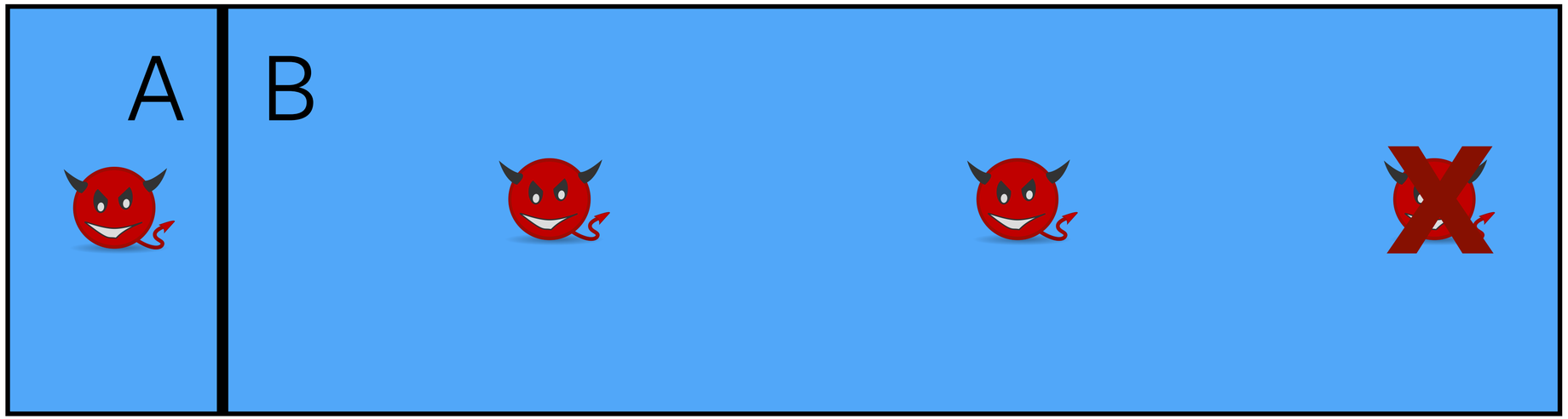
## Start with a small experiment




- Looks good so far....

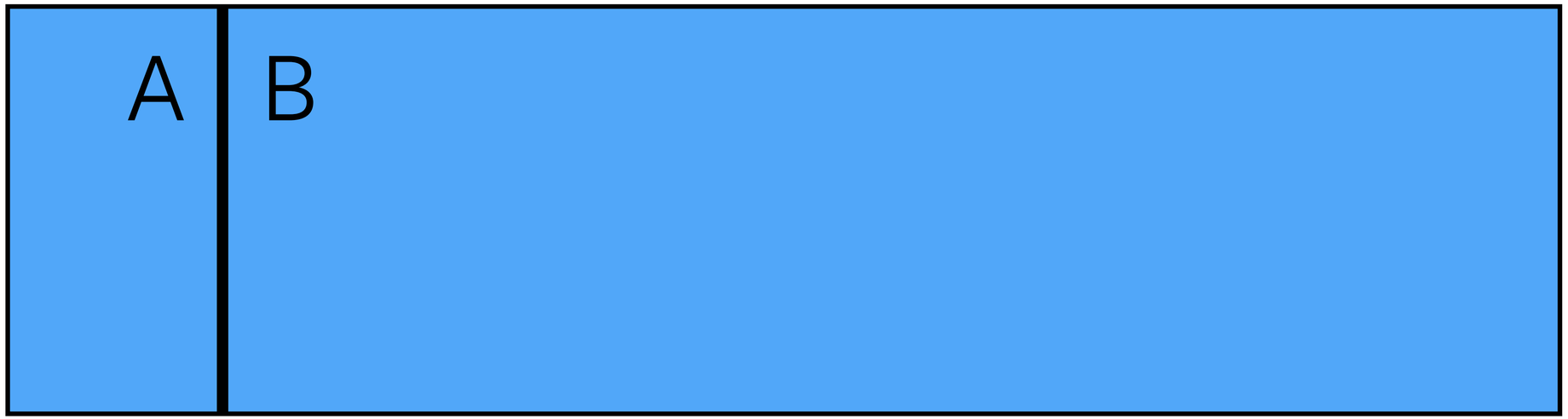
# The Perils of A/B Testing

## Roll it out to (almost) everyone — Option 1



# The Perils of A/B Testing

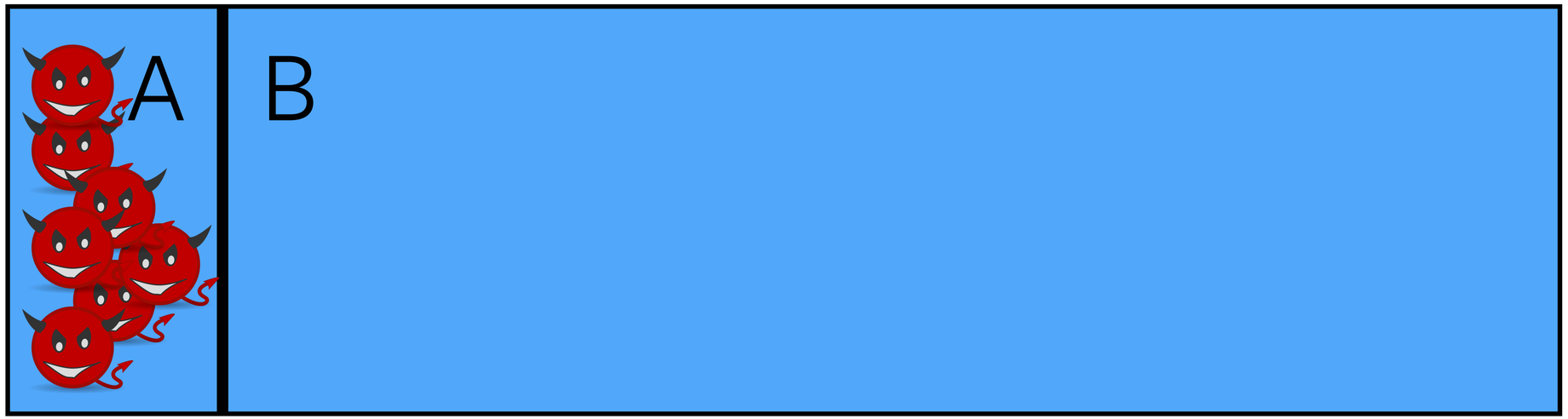
## Roll it out to (almost) everyone — Option 1




- Did the adversary give up or iterate?

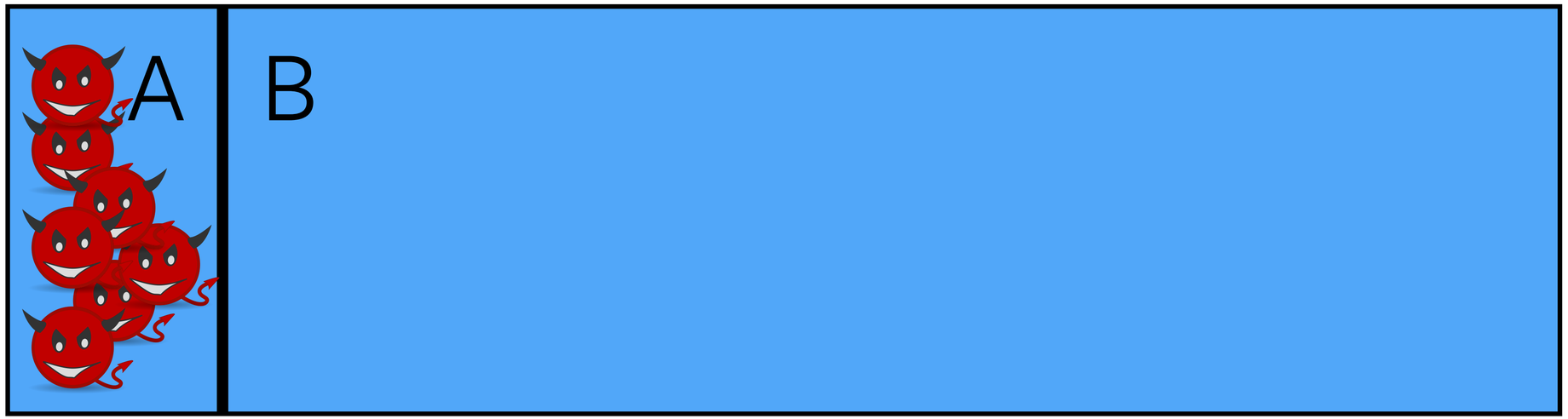
# The Perils of A/B Testing

## Roll it out to (almost) everyone — Option 2



# The Perils of A/B Testing

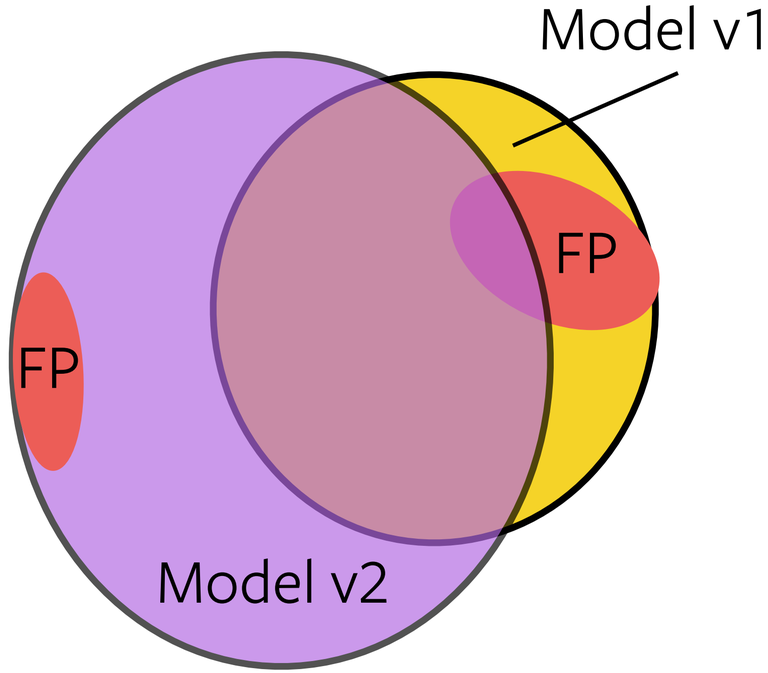
## Roll it out to (almost) everyone — Option 2




- Now your experiment is a vulnerability

# Using Shadow Mode

- Run new model online in "log-only" mode
- Evaluate performance where the new model *disagrees* with the old one.
- Push based on FP/FN tradeoff



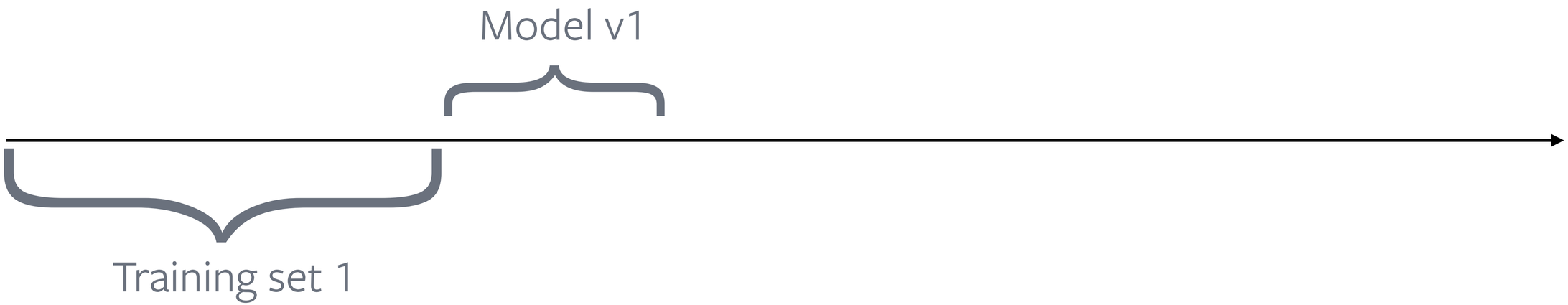
# Example II:
# Never Forget

# Refreshing your data

## Don’t forget the past!

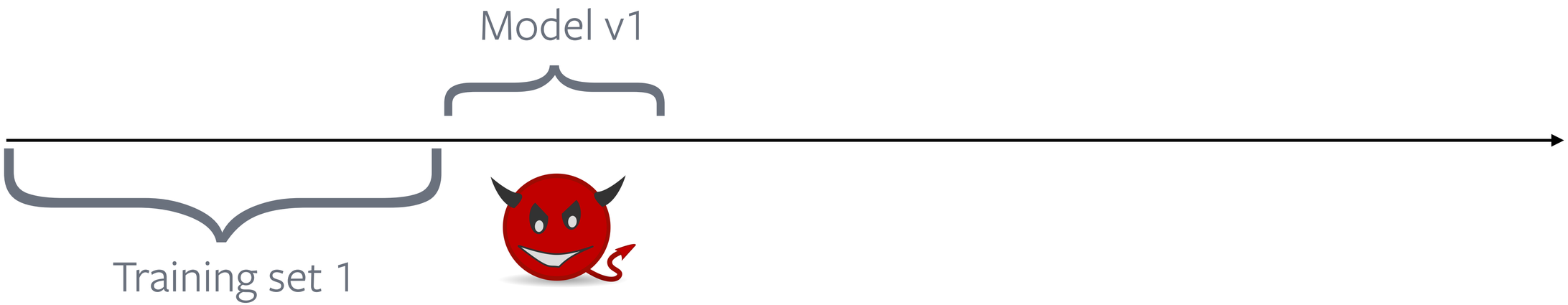
# Refreshing your data

## Don’t forget the past!



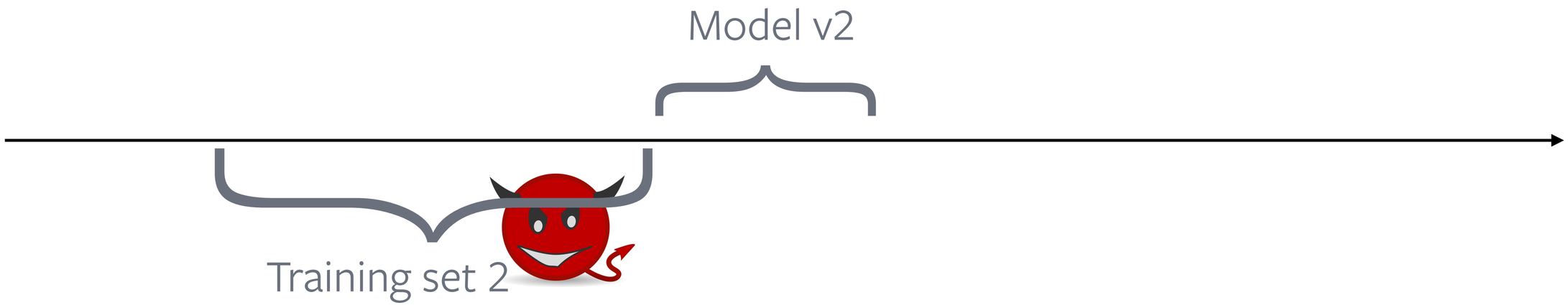
# Refreshing your data

## Don’t forget the past!



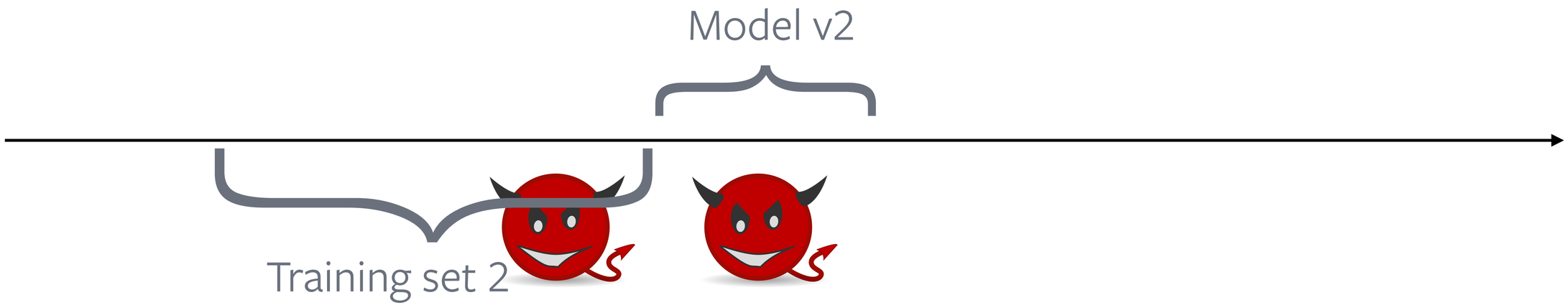
# Refreshing your data

## Don’t forget the past!



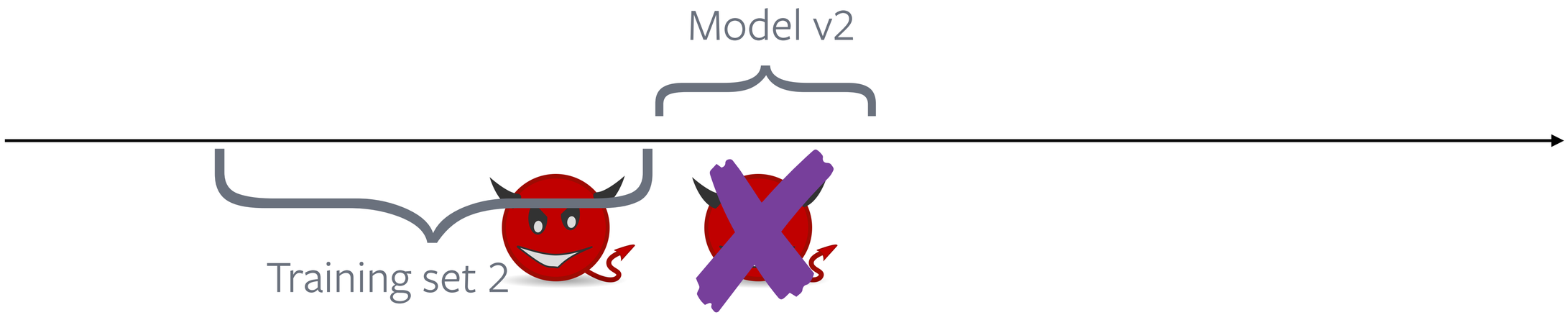
# Refreshing your data

## Don’t forget the past!



# Refreshing your data

## Don’t forget the past!



# Refreshing your data

## Don’t forget the past!

# Refreshing your data

## Don’t forget the past!

# Refreshing your data

## Don’t forget the past!


Mitigation:

- Keep old attacks around (exponential decay?)
- Keep old models around (raise thresholds?)

# Example III:
# It's a Race!

# Facebook has a few fake accounts

# Fake Accounts at Registration

Registration-time fake account classification has two fundamental problems:

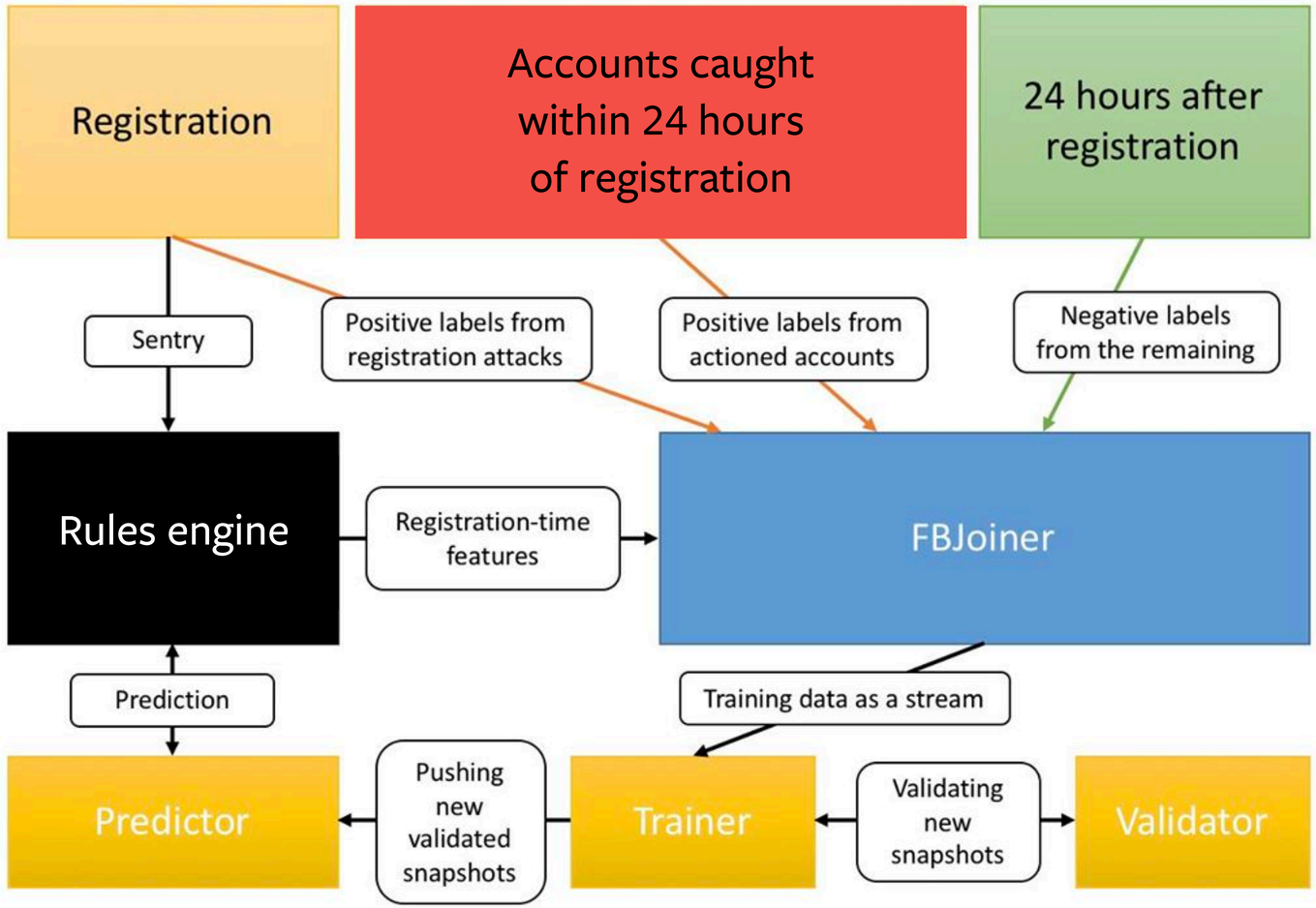
- Number of features is limited.
- Exposes a clear experimentation environment for attackers.

These two problems cause ML models to deteriorate fast.

- Each model iteration **requires significant manual work**
- **New model doesn’t learn what models in previous iterations learned**

# Solution 1: Learn Faster

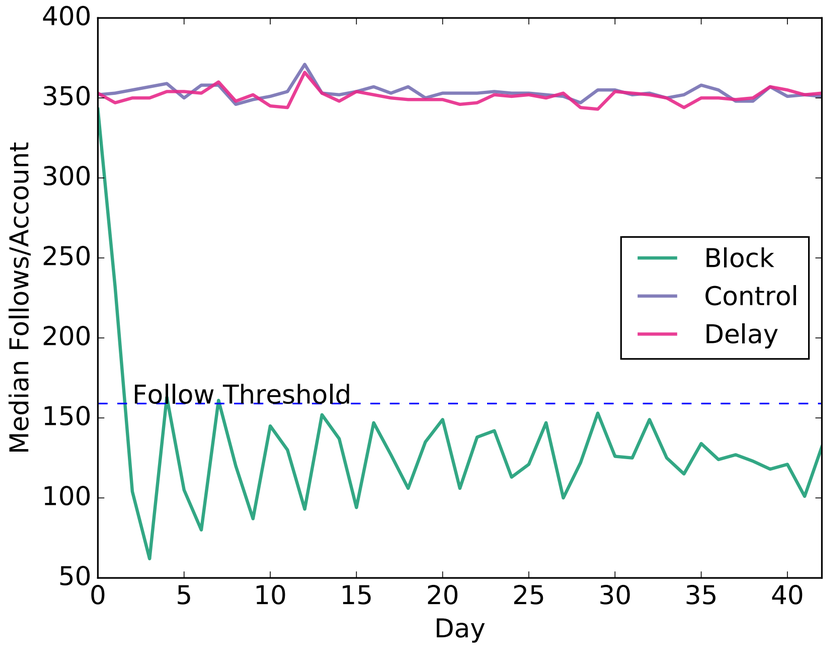
**Online Learning:**
Train a *single* model from *streaming* labels



# Solution 2: Act Slower

## Don’t give immediate feedback

- Introduce delay in blocking response (and/or)
- Undo the damage without telling the user.



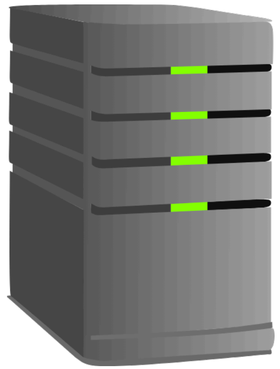
# Example IV:
# Don’t be Fooled

# What not to Do (I)

## Block on client-controlled signals

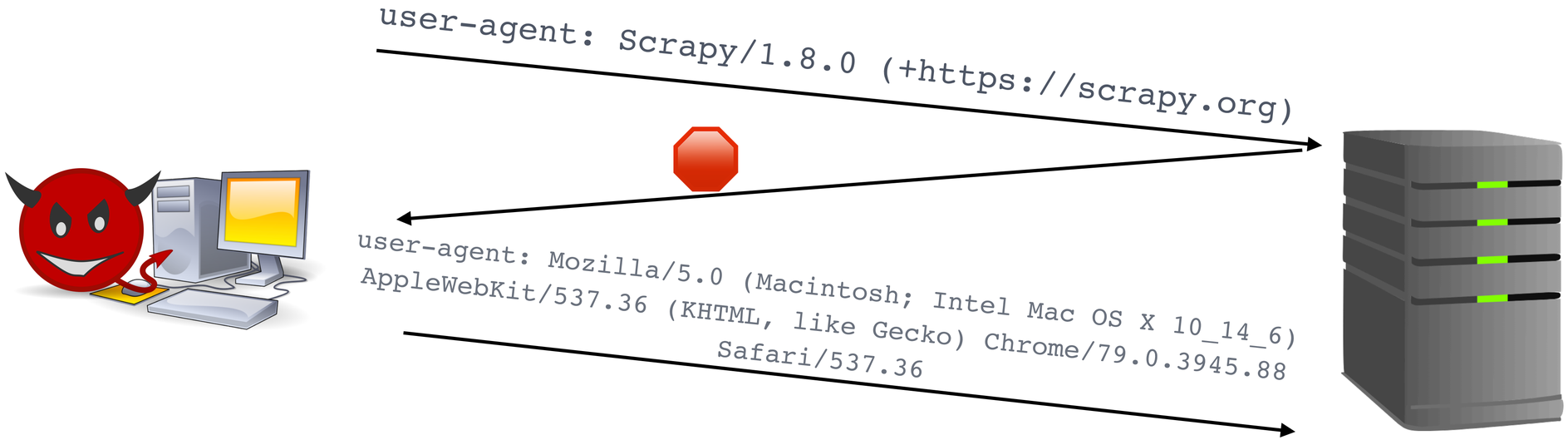
```
user-agent: Scrapy/1.8.0 (+https://scrapy.org)
```

```
user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88
Safari/537.36
```



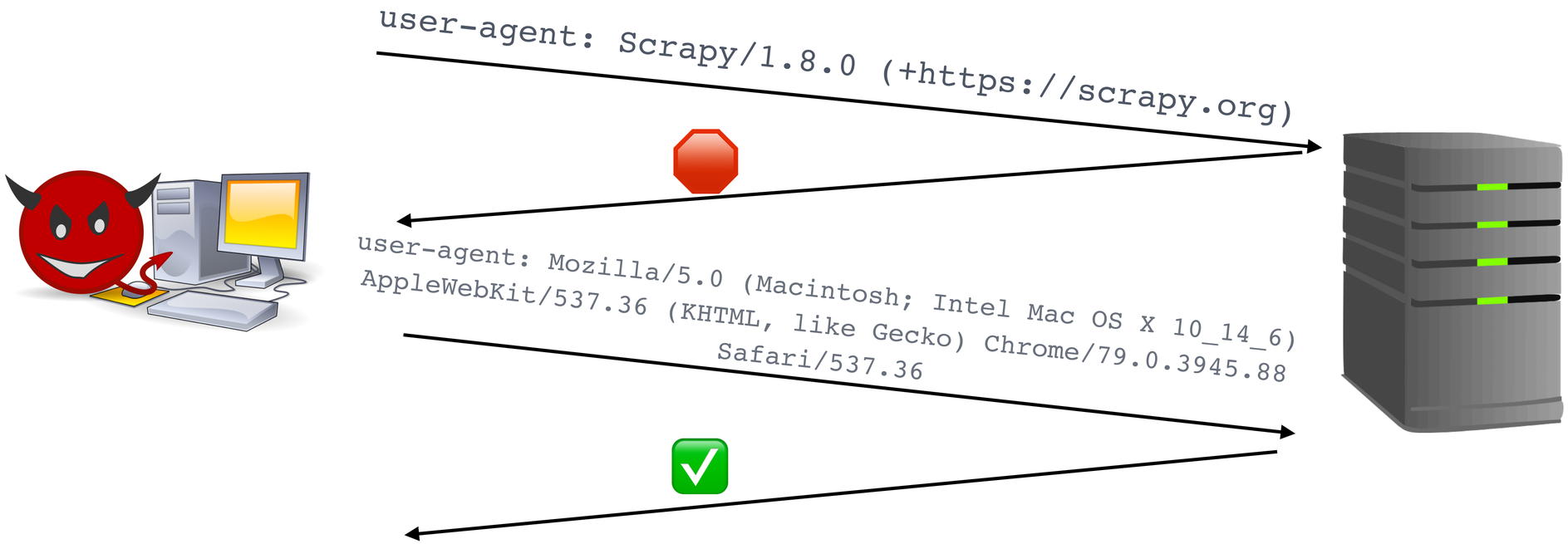
# What not to Do (I)

## Block on client-controlled signals



# What not to Do (I)

## Block on client-controlled signals



# What not to Do (II)

## Look for specific content to block

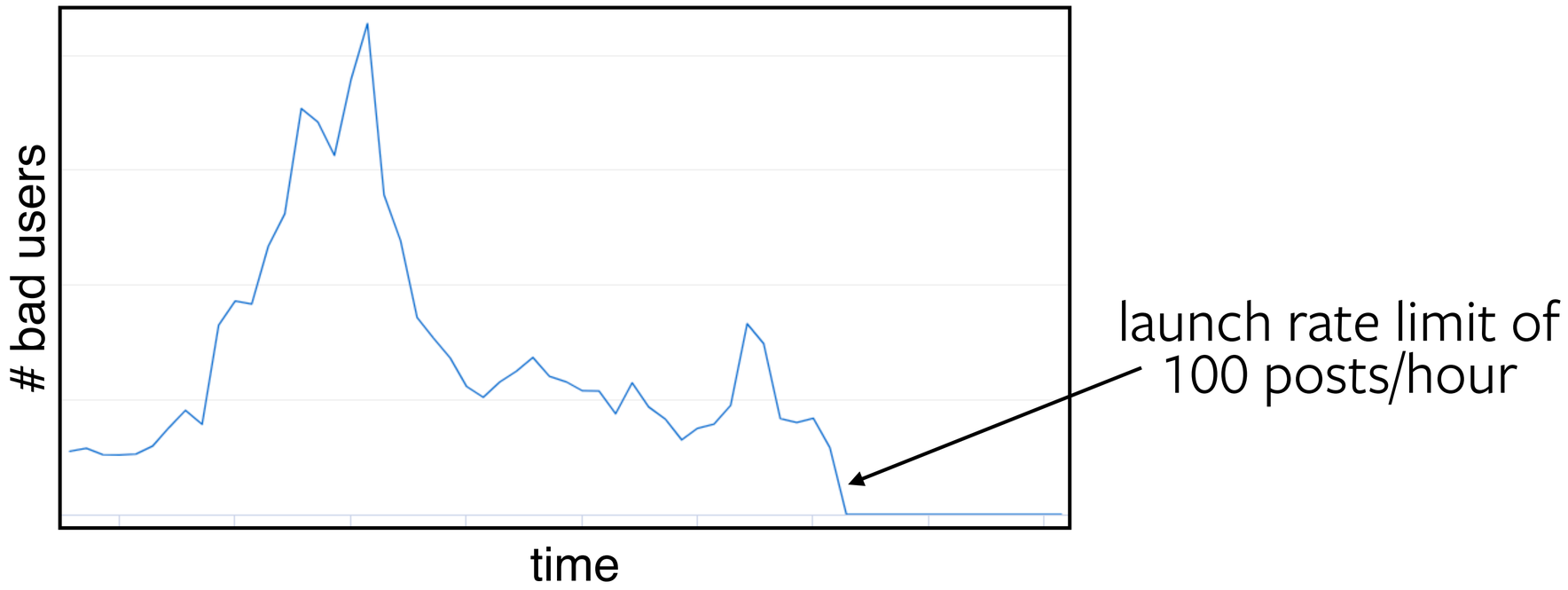



"We don't want to be the ones solving the CAPTCHAs"

# What not to do (III)

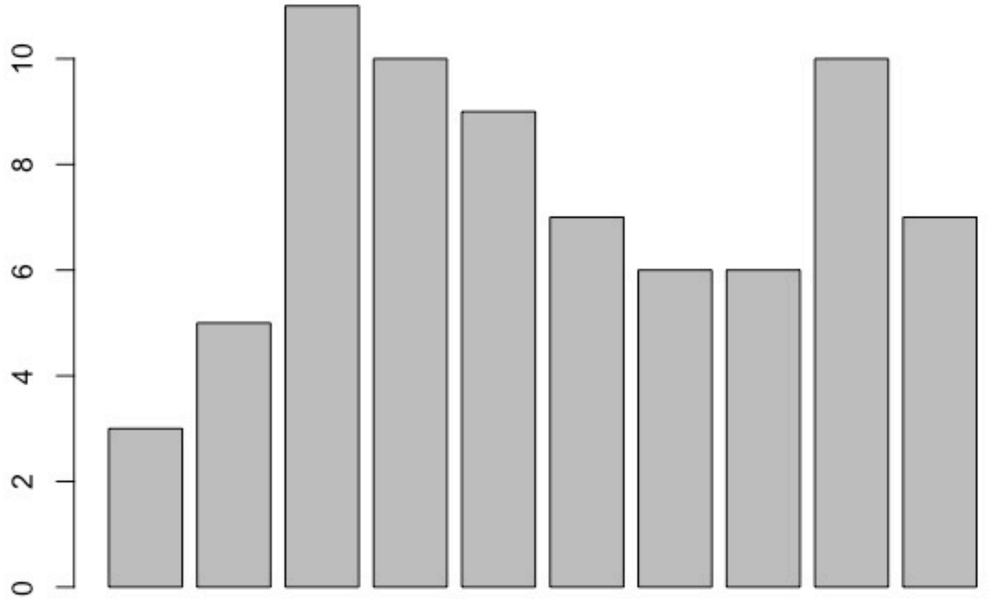
## Use the same signals for measurement and enforcement

Count users posting more than 100x/hour



# What to Do (I)

## Use data the adversary doesn't know/control

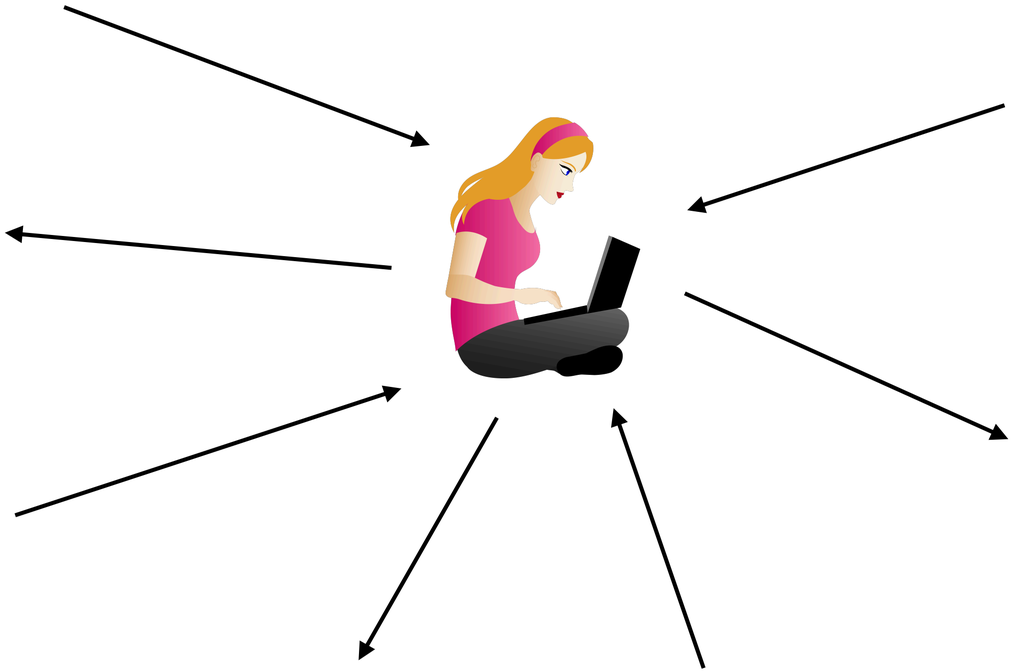


# What to Do (II)

## Focus on bad *behavior*, not only bad *content*

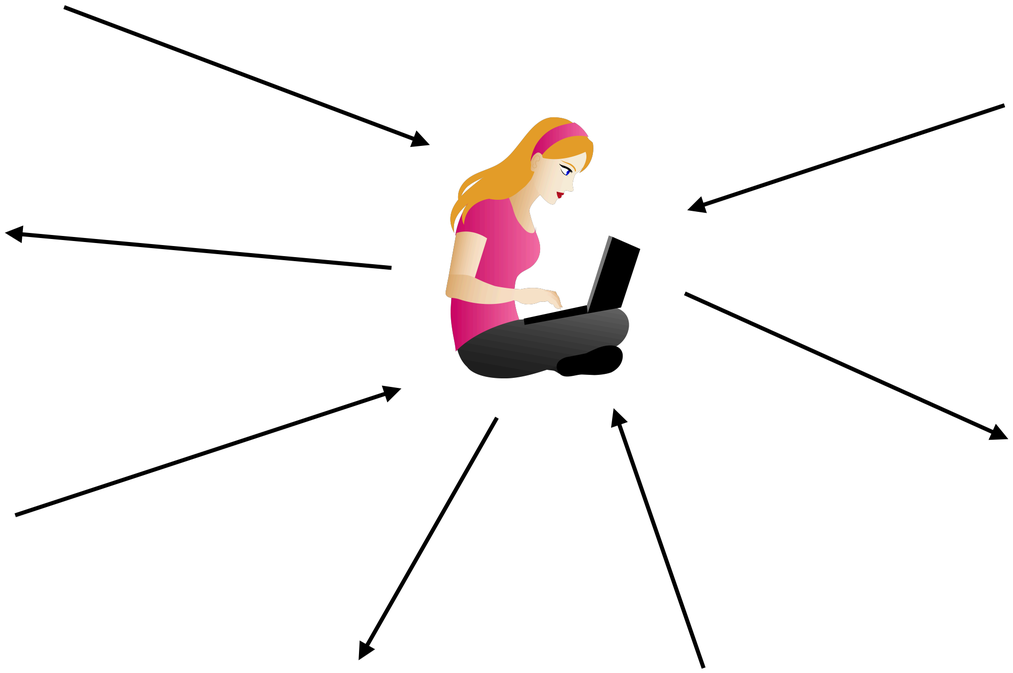
# What to Do (II)

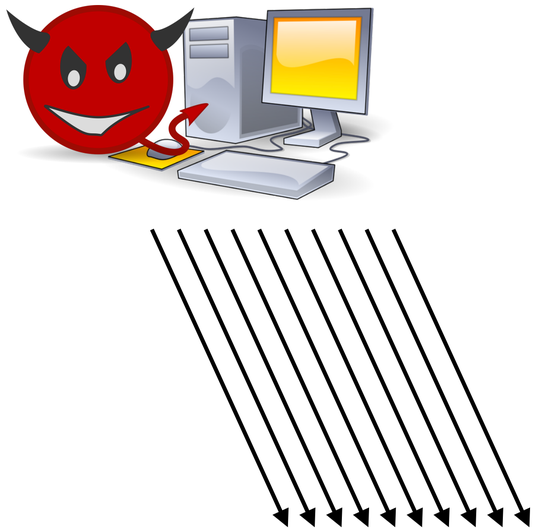
## Focus on bad *behavior*, not only bad *content*



# What to Do (II)

## Focus on bad *behavior*, not only bad *content*





# What to do (III)

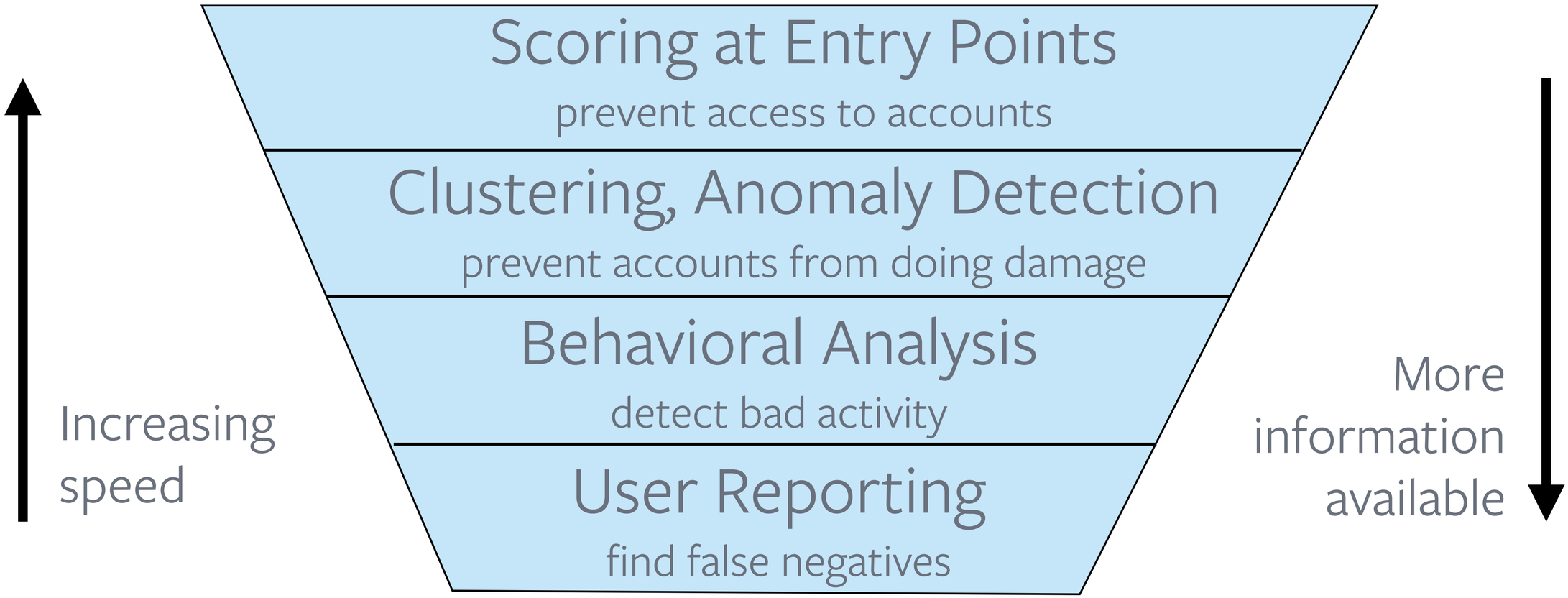
## Separate measurement and enforcement

At Facebook we split signals into two classes:

| Measurement | Enforcement |
| --- | --- |
| Network connection | Counts and rates |
| HTTP request | Graph relations |
| User-generated content | Activity sequence |

# Take-aways

# Set up Defense in Depth



# Open questions

- Can we combine online learning and active learning?
- How can we conduct rigorous A/B tests?
- What's the best way to avoid model forgetting?
- How do we prevent feedback between measurement (fragile signals) and detection (robust signals)?

# Thank you!

facebook

dfreeman@fb.com

*Want to help us?*
*Talk to me at the break!*