# Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail

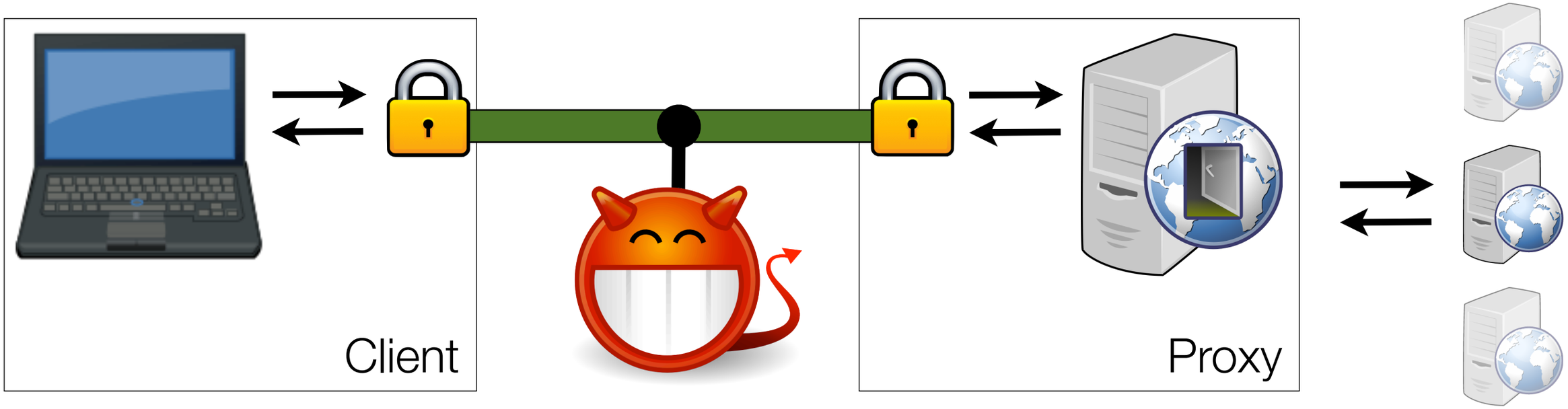
Kevin P Dyer
Portland State University

Joint work with:
**Scott Coull**, RedJack LLC
**Thomas Ristenpart**, University of Wisconsin-Madison
**Thomas Shrimpton**, Portland State University

# Peek-a-Boo, I Still See You:
# Why Efficient Traffic Analysis Countermeasures Fail...

*...to prevent website fingerprinting.*

The **client** makes a single request for a webpage over an encrypted link.




**Attacker's** goal is to **identify** the webpage requested.

**Security Intuition:**

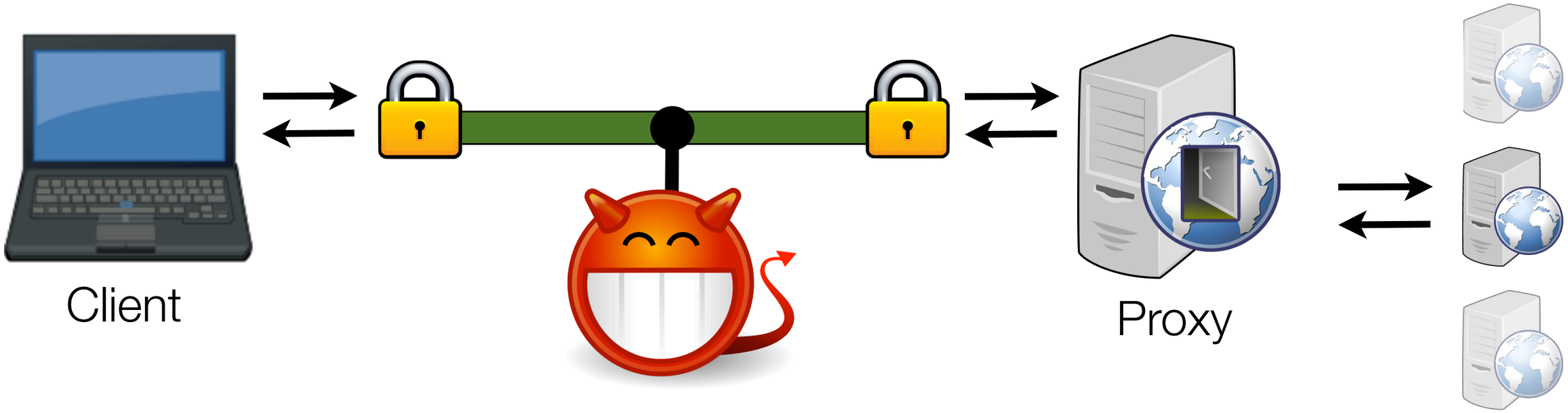
- only proxy's IP address revealed
- encryption hides everything else

# But

[Sun et al. ’02]
[Bissias et al. ‘05]
[Liberatore and Levine ’06]
[Herrmann et al. ’09]
[Wright et al. ’09]

[Lu et al. ’10]
[Chen et al. ’10]
[Luo et al. ’11]
[Panchenko et al. ’11]

# show otherwise

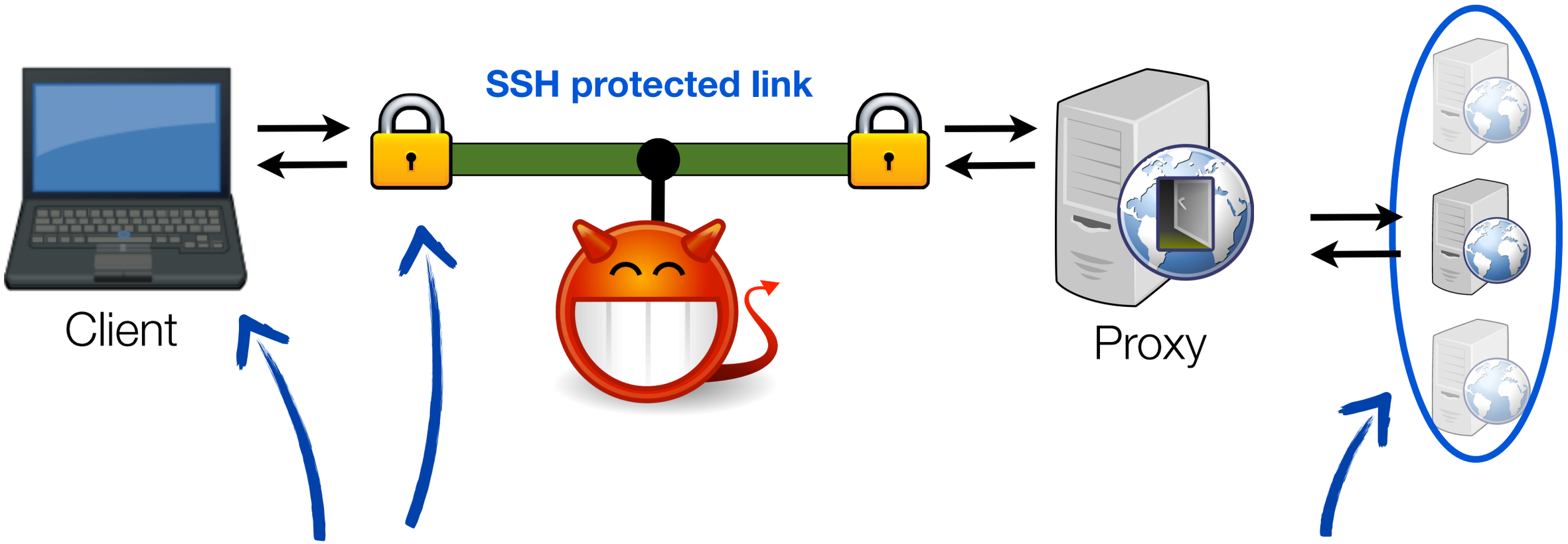



Attacker learns:

- packet lengths
- packet directions
- packet timings

} Enables **traffic analysis attacks**.

# [Liberatore and Levine ’06] Attack Scenario



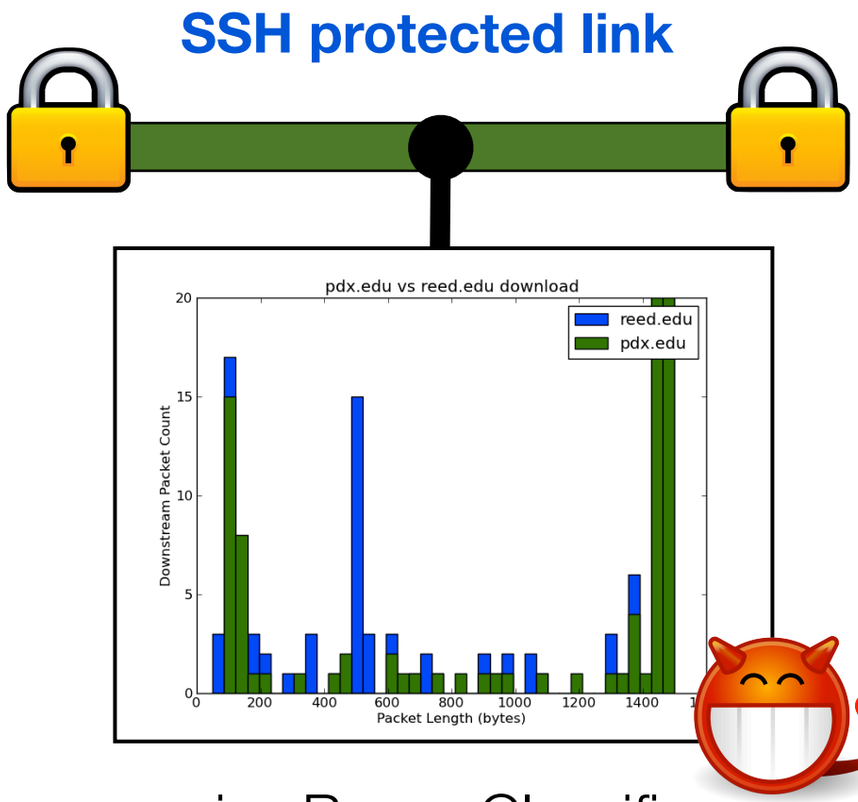

1. Attacker knows what client software is used.

2. Attacker knows the finite universe of webpages.

3. Attacker has labeled training data.

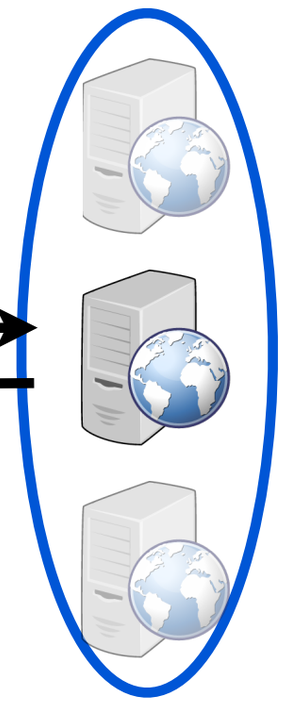
# [Liberatore and Levine ’06] Attack

Client




naive Bayes Classifier:
(packet direction, packet length) counts

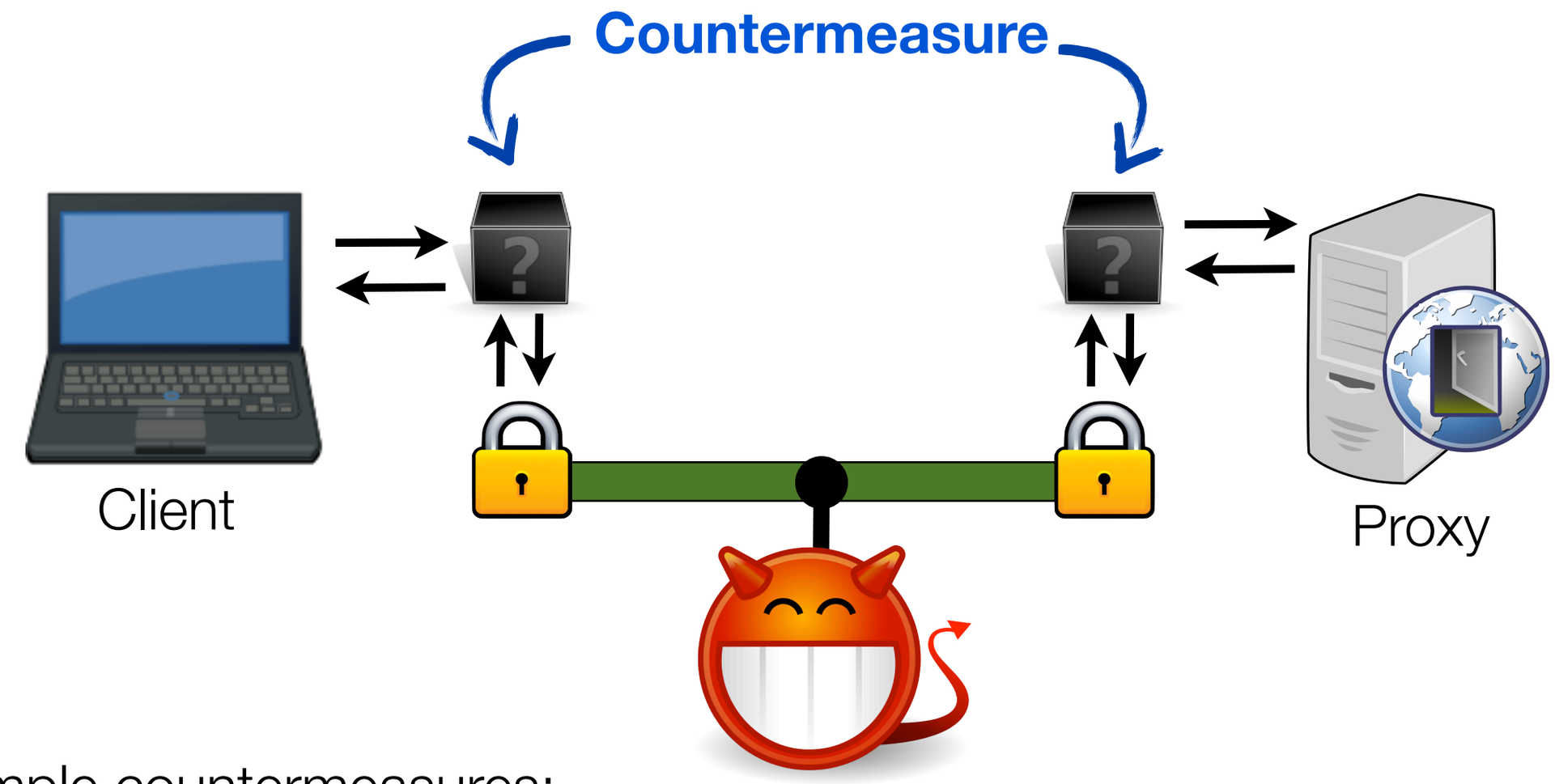
Proxy



**k=1000 webpages**

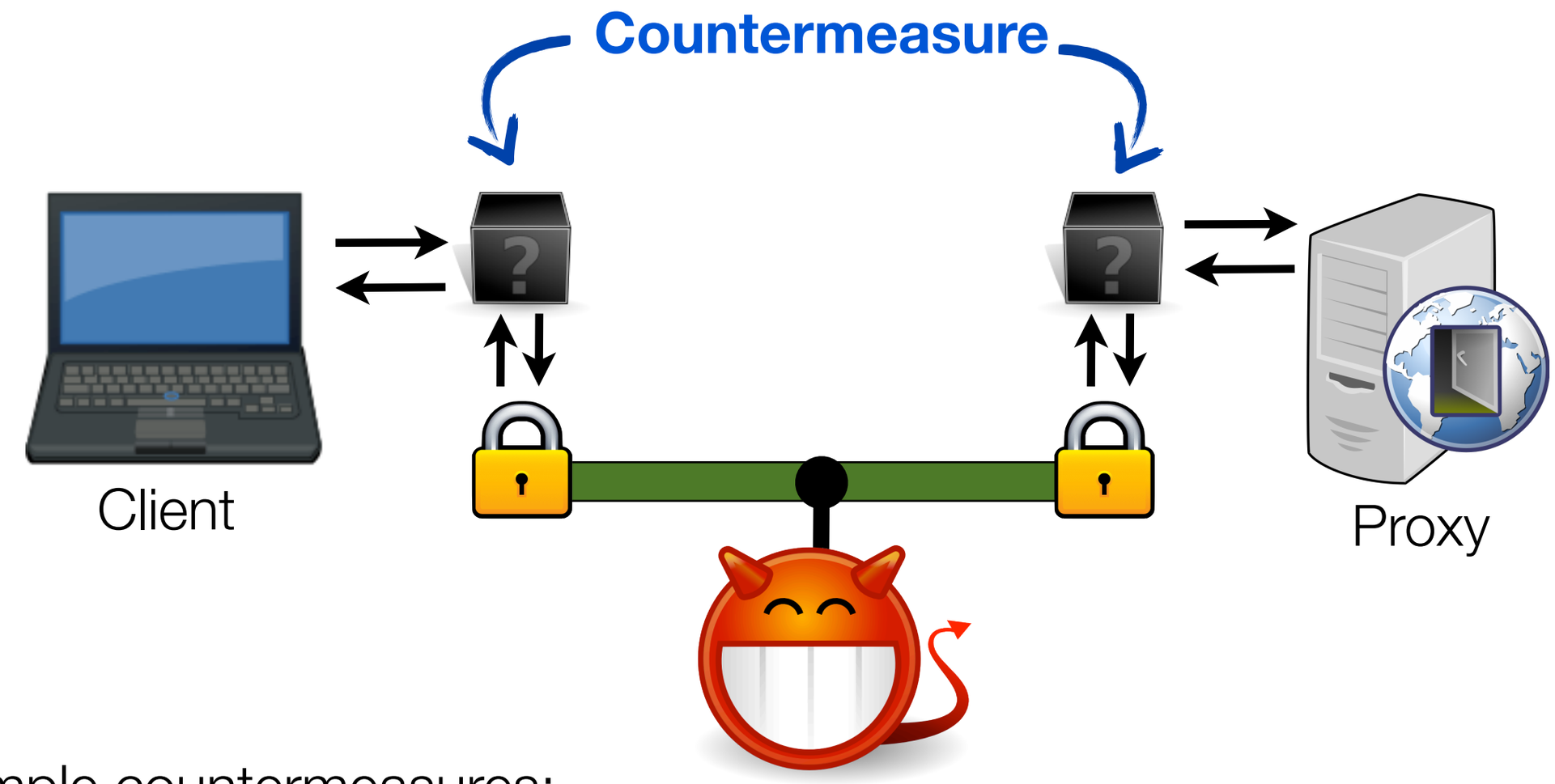
**Attacker can identify randomly chosen webpage with 68% accuracy!**

→ Packet lengths are a damaging side-channel




Example countermeasures:

- Pad to MTU
- Pad to random-length
- “Mice-elephants” padding
- Traffic Morphing [Wright et al. ’09]
- SSL RFC-compliant padding [SSL 3.0 RFC ’99]
- ...




Example countermeasures:

- Pad to MTU
- Pad to random-length
- “Mice-elephants” padding
- Traffic Morphing [Wright et al. ’09]
- SSL RFC-compliant padding [SSL 3.0 RFC ’99]
- ...

Do these countermeasures prevent TA attacks?

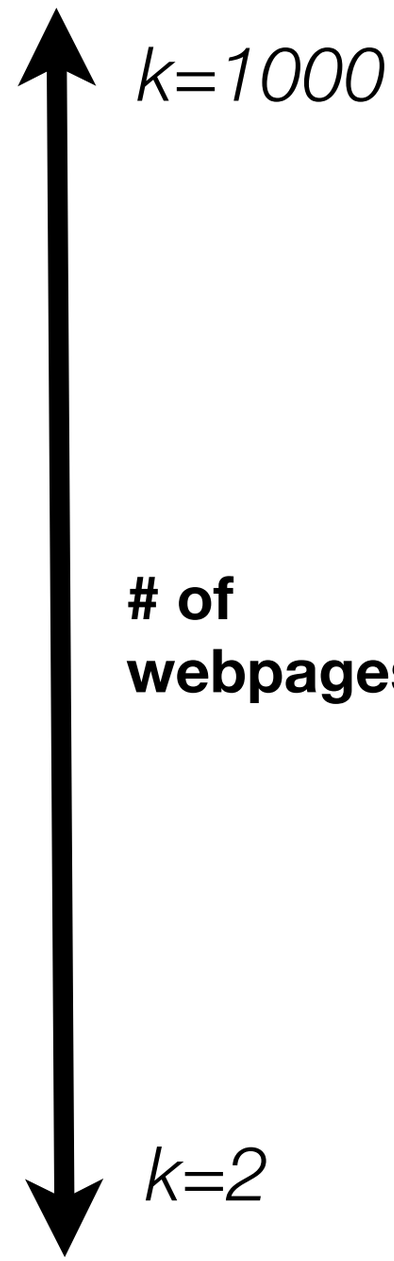
# Prior work does not provide a clear answer

No Countermeasure

68% [LL]

Pad to MTU

8% [LL]



# Prior work does not provide a clear answer

No Countermeasure

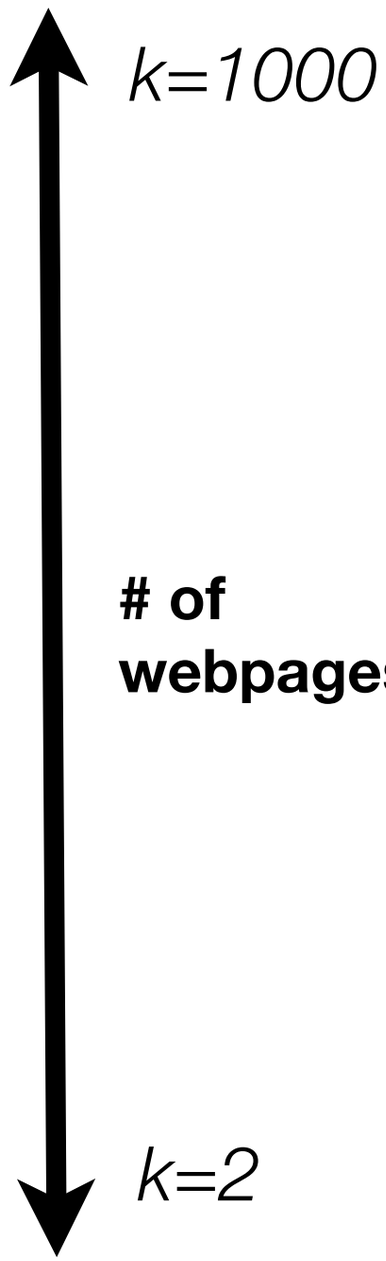
68% [LL]

98% [W]

Pad to MTU

8% [LL]

86% [W]



# Prior work does not provide a clear answer

No Countermeasure

68% [LL]

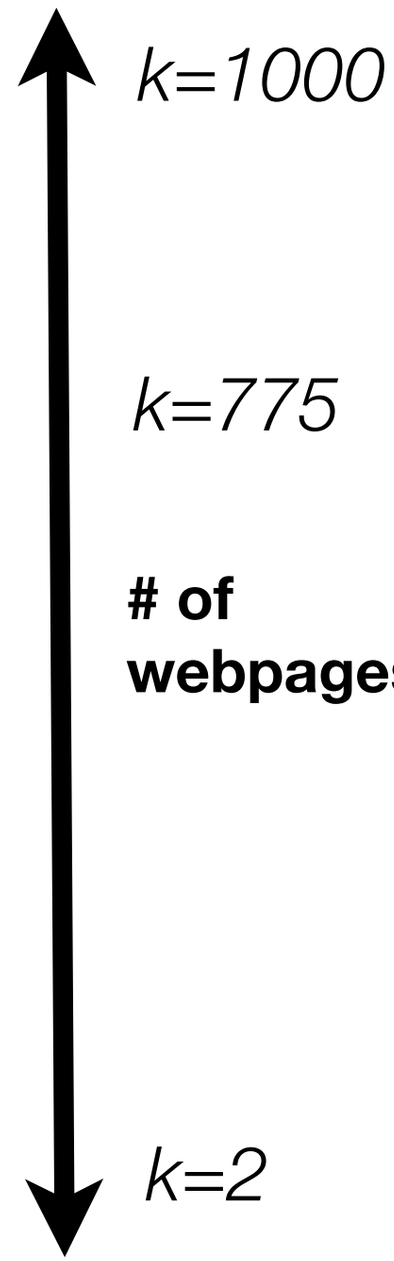
98% [H]

98% [W]

Pad to MTU

8% [LL]

86% [W]



# Prior work does not provide a clear answer

No Countermeasure

68% [LL]

98% [H]

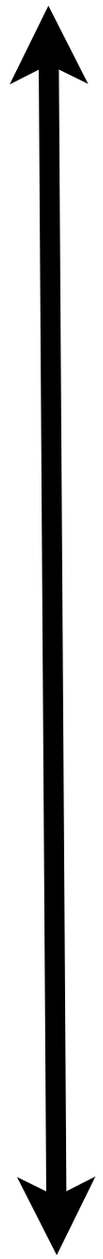
98% [W]

Pad to MTU

8% [LL]

86% [W]

**What about other values of *k*?**



# Prior work does not provide a clear answer

No Countermeasure

68% [LL]

98% [H]

**Does the data set used impact efficacy?**

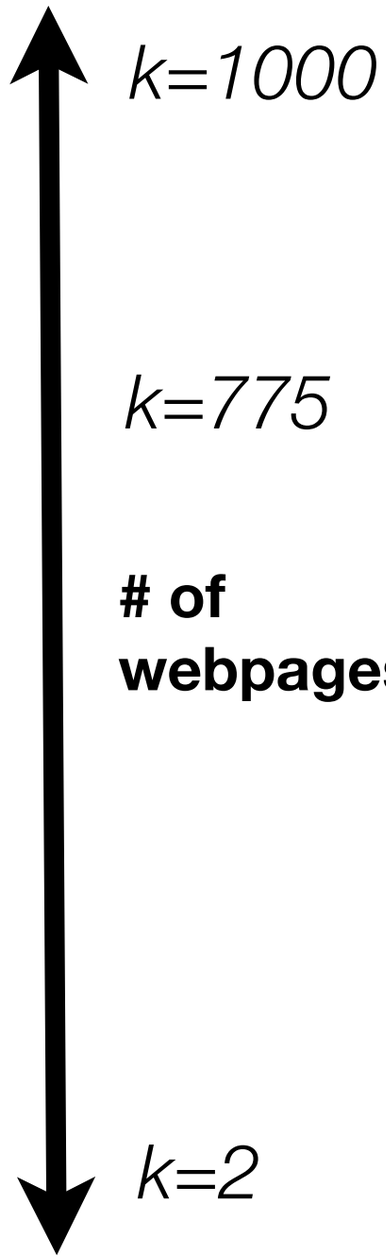
98% [W]

Pad to MTU

8% [LL]

86% [W]

**What about other values of *k*?**



# Prior work does not provide a clear answer

No Countermeasure

68% [LL]

98% [H]

Pad to MTU

8% [LL]

**What about other classification strategies?**

**Does the data set used impact efficacy?**

98% [W]

86% [W]

**What about other values of *k*?**

*k=1000*

*k=775*

**# of webpages**

*k=2*

# Prior work does not provide a clear answer

No Countermeasure

68% [LL]

98% [H]

98% [W]

Pad to MTU

8% [LL]

86% [W]

**What about other classification strategies?**

**Does the data set used impact efficacy?**

**What about other countermeasures?**

**What about other values of *k*?**

*k=1000*

*k=775*

*k=2*

**# of webpages**

# Our work

1. Comprehensive evaluation of traffic analysis countermeasures.

   No countermeasure works in the LL setting.

2. In-depth analysis of traffic features

   Coarse features (e.g., time, bandwidth) enable high-accuracy attacks despite countermeasures

# Our work

1. Comprehensive evaluation of traffic analysis countermeasures.

   No countermeasure works in the LL setting.

2. In-depth analysis of traffic features

   Coarse features (e.g., time, bandwidth) enable high-accuracy attacks despite countermeasures

Pessimistic conclusion:
*efficient* countermeasures can't hide "coarse" features.

# Our Comprehensive Analysis

**9 countermeasures**

5 padding schemes
2 TLS/SSH “inspired” padding schemes
2 versions of traffic morphing

**6 classifiers**

[Liberatore and Levine] **naive Bayes, Jaccard**
[Wright et al.] **naive Bayes**
[Lu et al.] **edit distance**
[Herrmann et al.] **multinomial naive-Bayes**
[Panchenko et al.] **support vector machine**

**10 “universe” sizes**

k=2,4,8,16,32,64,128,256,512,775

**2 data sets**

Liberatore and Levine (2000 websites)
Herrmann et al. (775 websites)

# The countermeasures

- Session Random 255
- Packet Random 255
- Linear Padding
- Exponential Padding
- Mice-Elephants Padding
- Pad to MTU
- Packet Random MTU
- Traffic Morphing
- Direct Target Sampling

# The countermeasures

- Session Random 255
- Packet Random 255
- Linear Padding
- Exponential Padding
- Mice-Elephants Padding
- **Pad to MTU**
- Packet Random MTU
- Traffic Morphing
- Direct Target Sampling

**Every packet on the wire is padded to a fixed length.**

# The countermeasures

- Session Random 255
- Packet Random 255
- Linear Padding
- Exponential Padding
- Mice-Elephants Padding
- **Pad to MTU**
- Packet Random MTU
- **Traffic Morphing**
- Direct Target Sampling

**Every packet on the wire is padded to a fixed length.**

**[Wright et al. ’09]**
**- Pads packets**
**- Chops packets**
**- Sends dummy packets**
**- Mimics packet-length distributions**

# Some representative results

**Classifier accuracy at *k*=512**

| | None | Pad to MTU | Traffic Morphing |
|---|---|---|---|
| Herrmann et al. | 99% | 2% | 3% |
| Liberatore and Levine | 97% | 41% | 17% |
| Panchenko et al. | 96% | 82% | 81% |

# Some representative results

## Classifier accuracy at *k*=512

| | None | Pad to MTU | Traffic Morphing |
|---|---|---|---|
| Herrmann et al. | **99%** | 2% | 3% |
| Liberatore and Levine | 97% | 41% | 17% |
| Panchenko et al. | 96% | 82% | 81% |

**Best performer with no countermeasure applied.**

# Some representative results

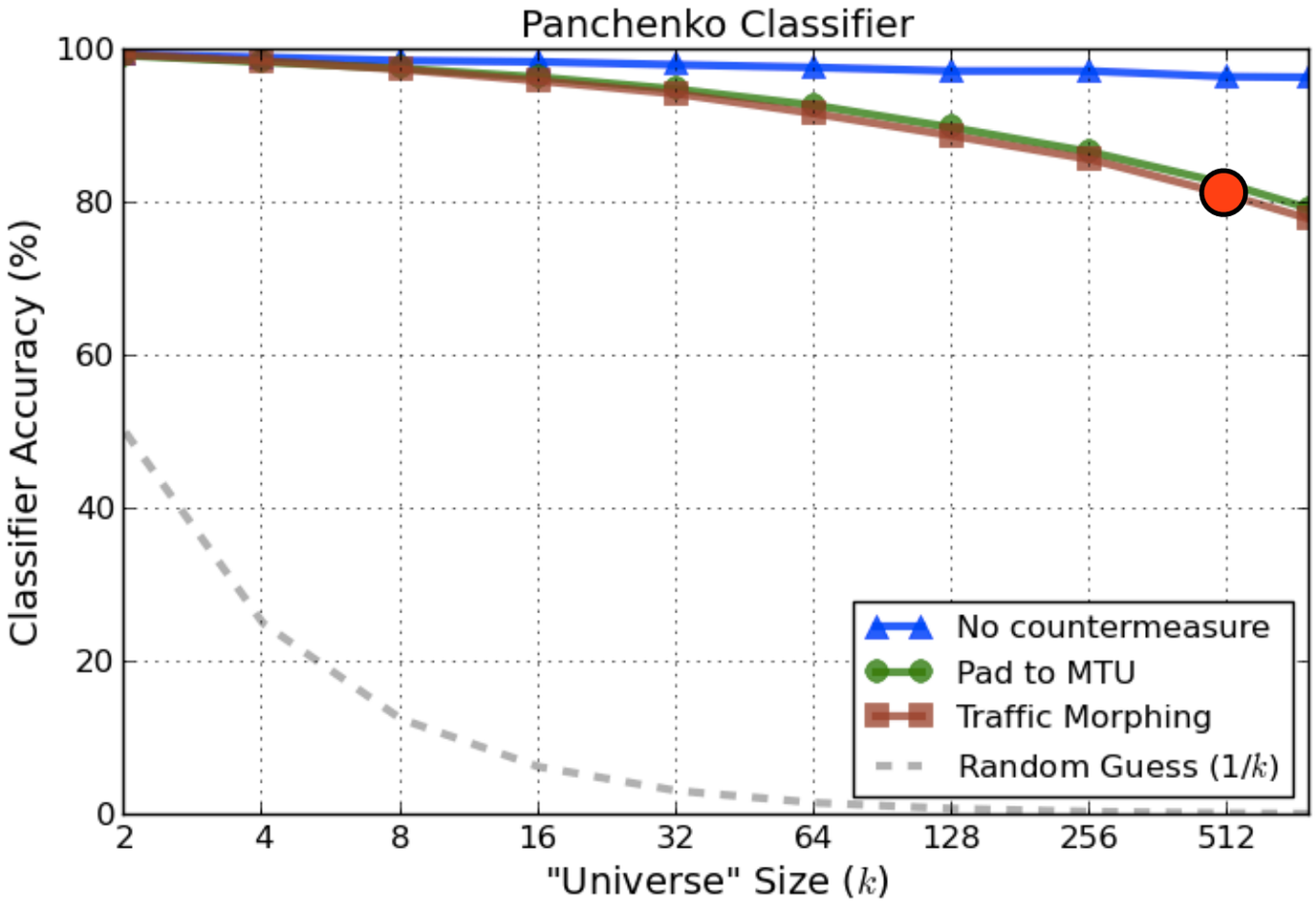
## Classifier accuracy at *k*=512

| | None | Pad to MTU | Traffic Morphing |
|---|---|---|---|
| Herrmann et al. | **99%** | 2% | 3% |
| Liberatore and Levine | 97% | 41% | 17% |
| Panchenko et al. | 96% | **82%** | **81%** |

**Best performer with no countermeasure applied.**

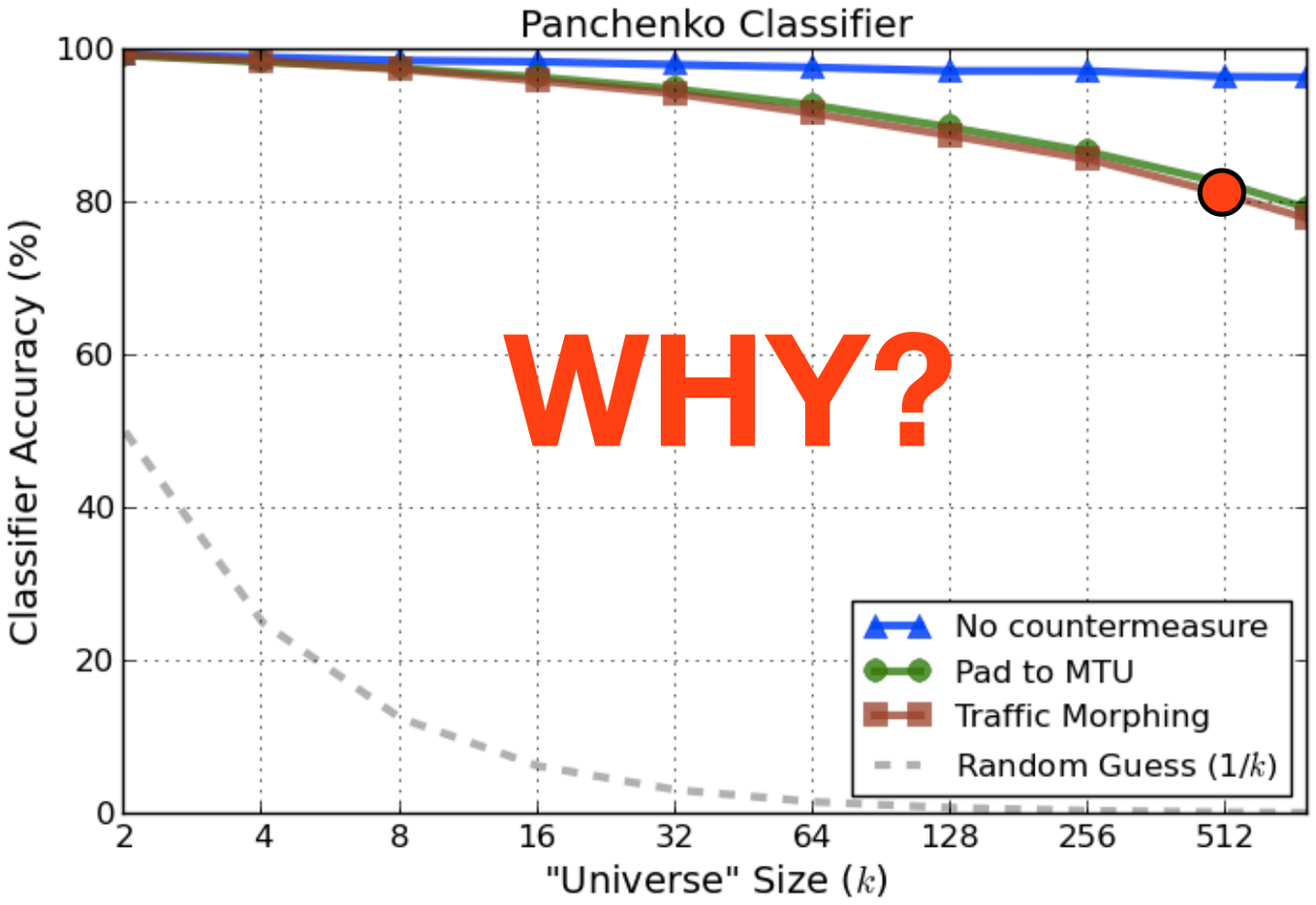
**Best performer with countermeasures applied.**

# Under the hood of the [Panchenko ’11] classifier




- Pad to MTU **82%** at k=512
  Traffic Morphing **81%** at k=512

# Under the hood of the [Panchenko ’11] classifier



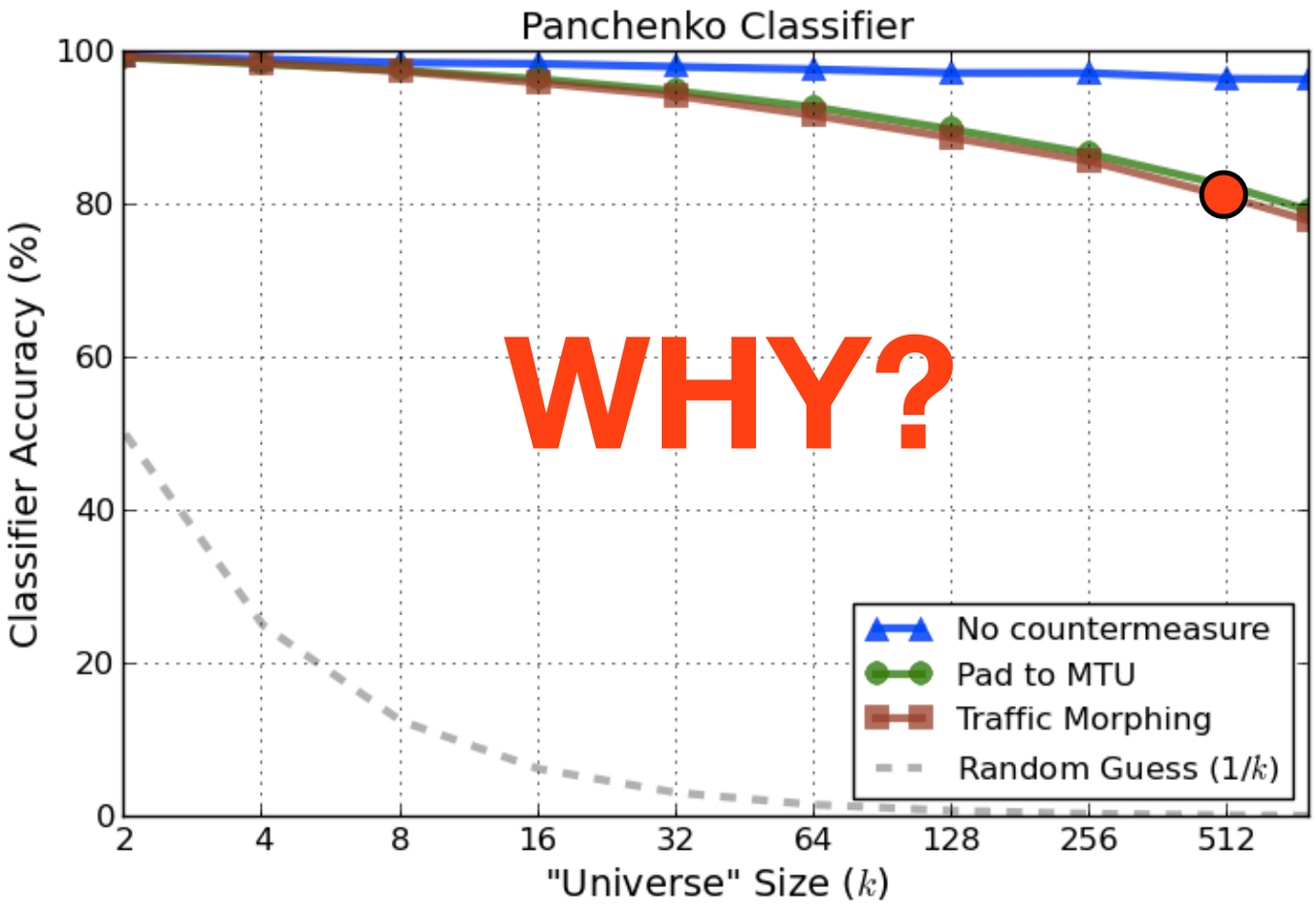

- Pad to MTU **82%** at k=512
  Traffic Morphing **81%** at k=512

Support vector machine

Features used:

- Packet lengths upstream
- Packet lengths downstream
- Burst bandwidth upstream
- Burst bandwidth downstream
- HTML marker downstream
- Number markers upstream
- Number markers downstream
- Total bytes transmitted upstream
- Total bytes transmitted downstream
- Percentage of downstream packets
- Total number of packets upstream
- Total number of packets downstream
- Occurring packet lengths downstream
- Occurring packet lengths upstream

# Under the hood of the [Panchenko ’11] classifier




- Pad to MTU **82%** at k=512
  Traffic Morphing **81%** at k=512

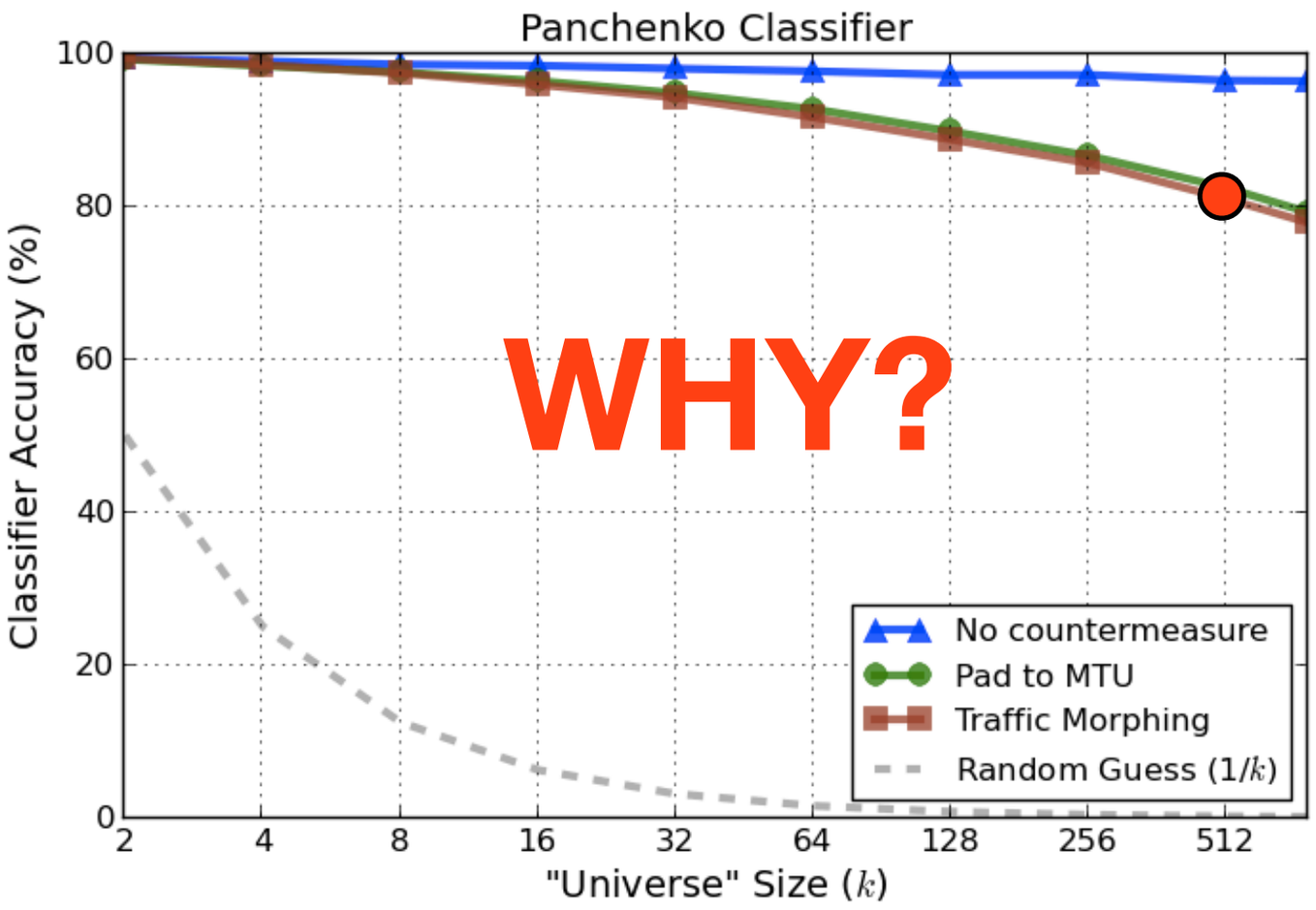
Support vector machine X

Features used:

- Packet lengths upstream
- Packet lengths downstream
- Burst bandwidth upstream
- Burst bandwidth downstream
- HTML marker downstream
- Number markers upstream
- Number markers downstream
- Total bytes transmitted upstream
- Total bytes transmitted downstream
- Percentage of downstream packets
- Total number of packets upstream
- Total number of packets downstream
- Occurring packet lengths downstream
- Occurring packet lengths upstream

?

# Under the hood of the [Panchenko ’11] classifier




- Pad to MTU **82%** at k=512
- Traffic Morphing **81%** at k=512

Support vector machine X

Features used:

- ~~Packet lengths upstream~~
- ~~Packet lengths downstream~~
- Burst bandwidth upstream
- Burst bandwidth downstream
- HTML marker downstream
- Number markers upstream
- Number markers downstream
- Total bytes transmitted upstream
- Total bytes transmitted downstream
- Percentage of downstream packets
- Total number of packets upstream
- Total number of packets downstream
- ~~Occurring packet lengths downstream~~
- ~~Occurring packet lengths upstream~~

?

# Digging deeper: Understanding the features

1. Identify “coarse” feature.

   Time  Bandwidth  Burst Bandwidth

2. Implement a feature-specific classifier.

3. Run classifier against all countermeasures.

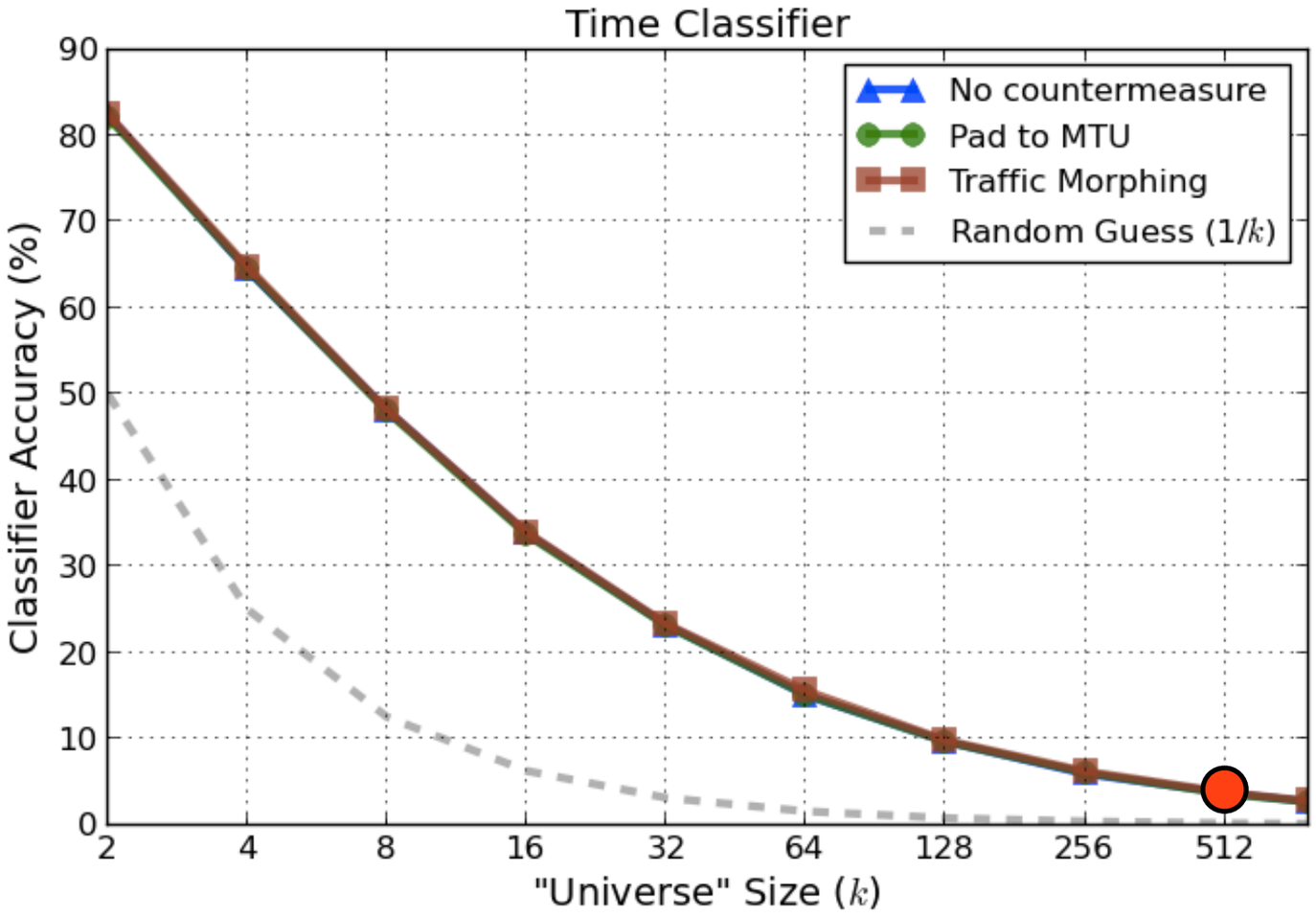
# “Coarse” Traffic Features with Pad to MTU



| | None | Pad to MTU |
|---|---|---|
| time | 2.8s | **2.8s** |
| bandwidth | 277KB | **347KB** |
| bursts | 13 | **13** |

| | None | Pad to MTU |
|---|---|---|
| time | 5.2s | **5.2s** |
| bandwidth | 1794KB | **2560KB** |
| bursts | 107 | **107** |

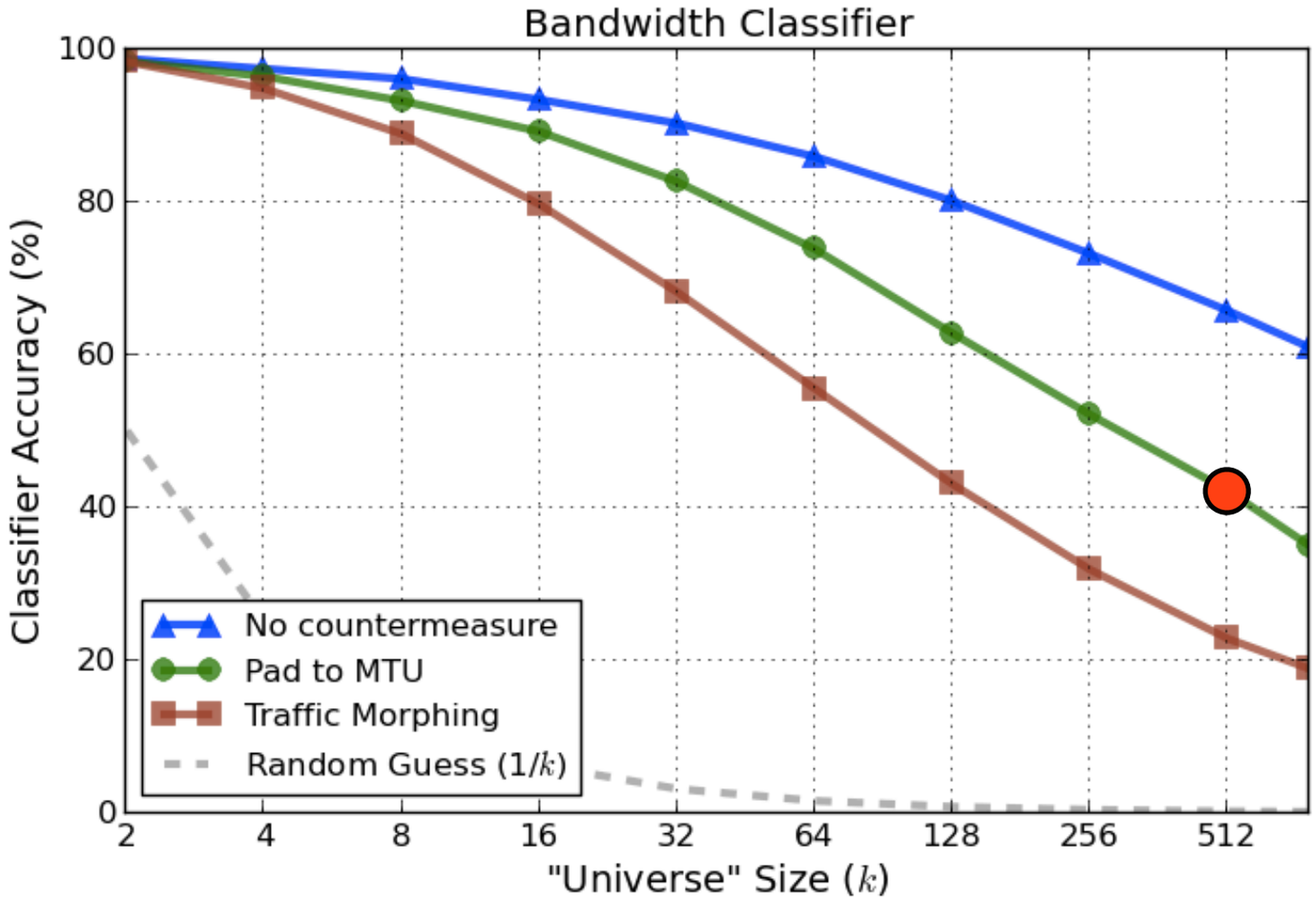
# Feature: Time Elapsed




Useful for small values of *k*

- “Pad to MTU” **5%** at k=512

# Feature: Bandwidth

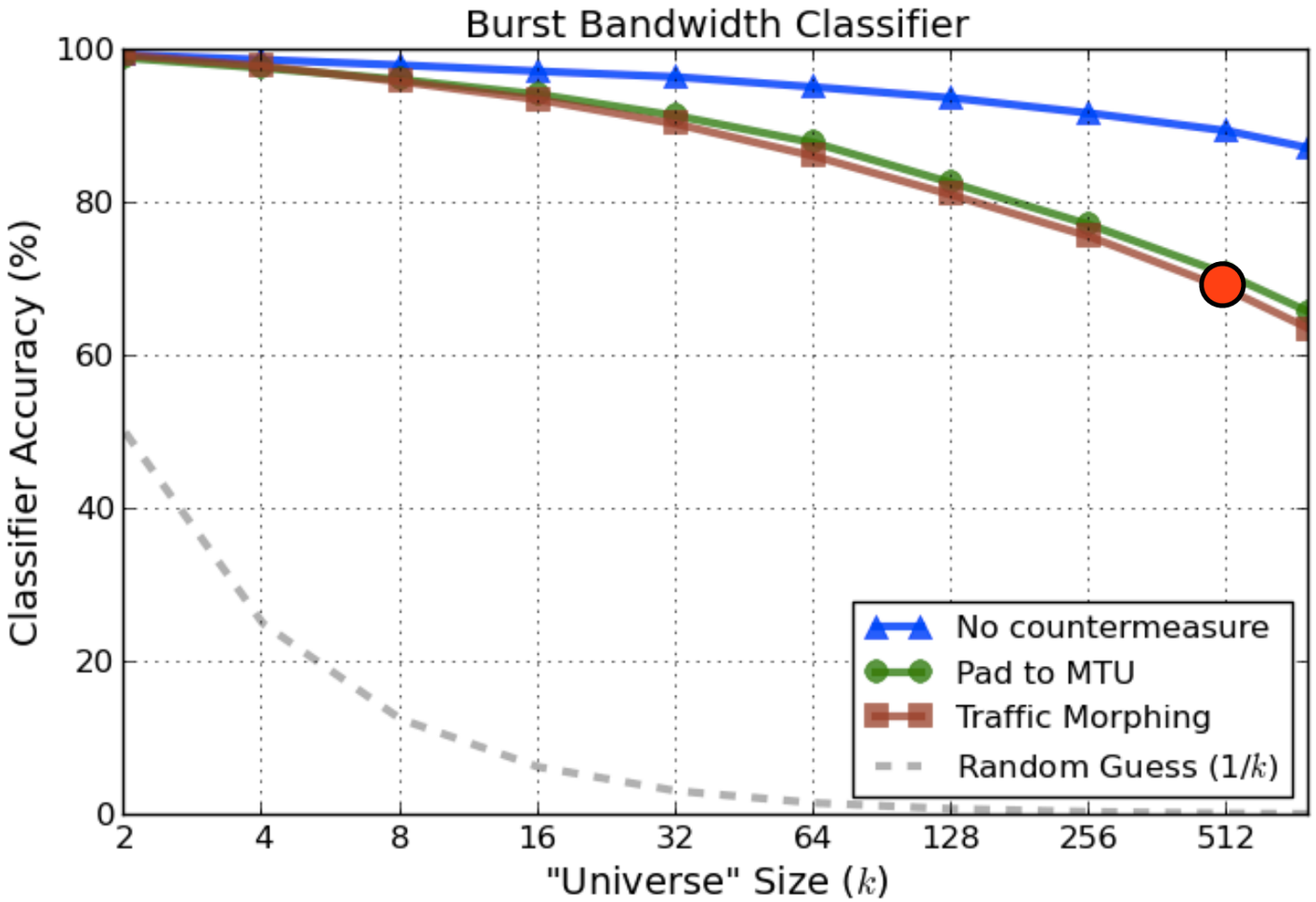



More robust to large values *k* than the time classifier

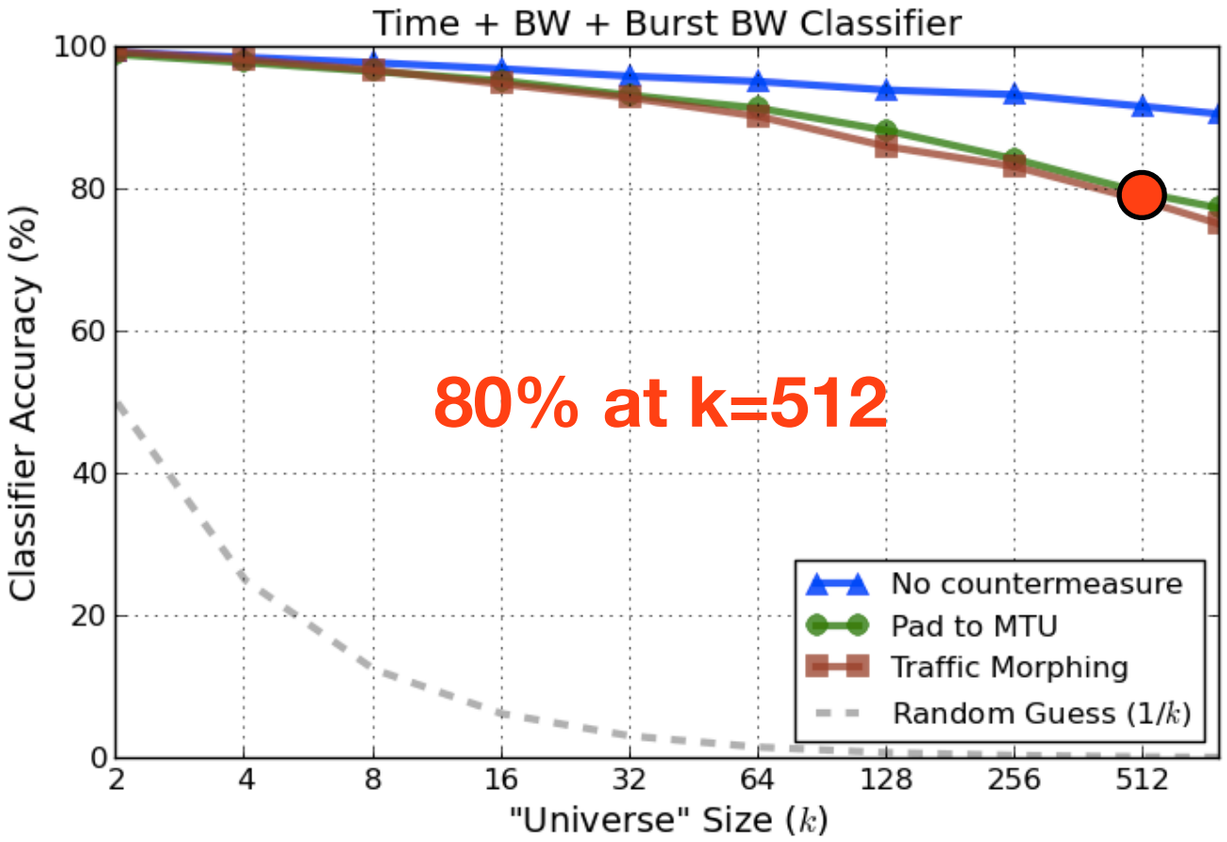
Still a “coarse” measurement

- “Pad to MTU” **42%** at k=512

# Feature: Burst Bandwidth

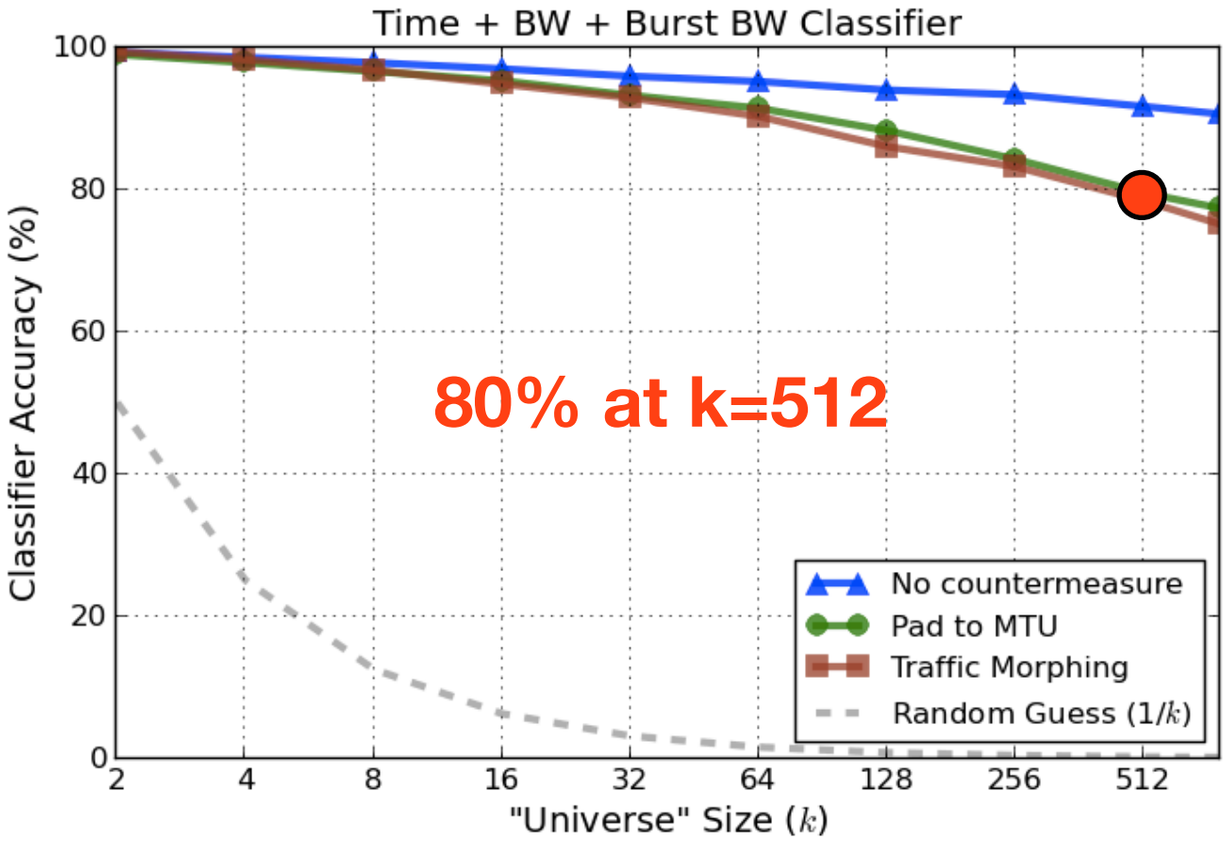


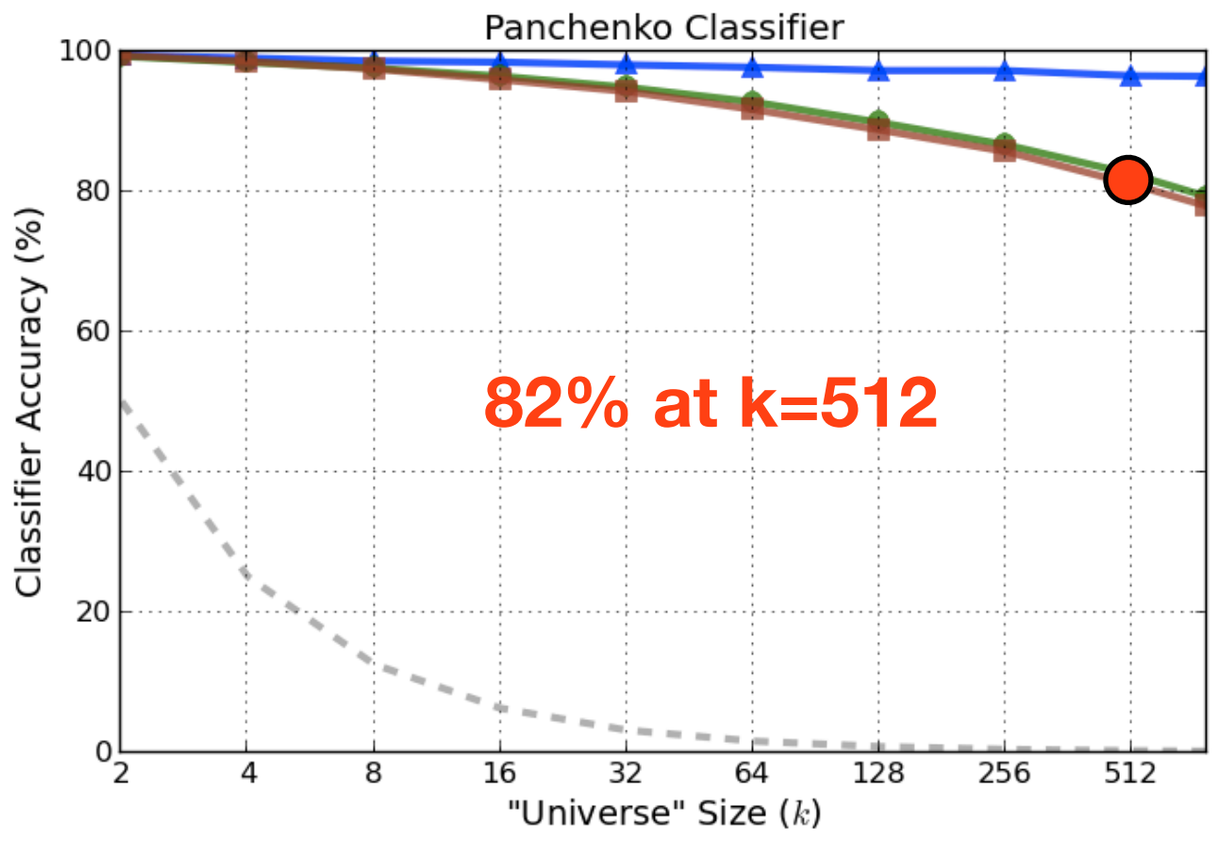

- “Pad to MTU” **71%** at k=512




Putting coarse features together:
simple naive Bayes classifier using

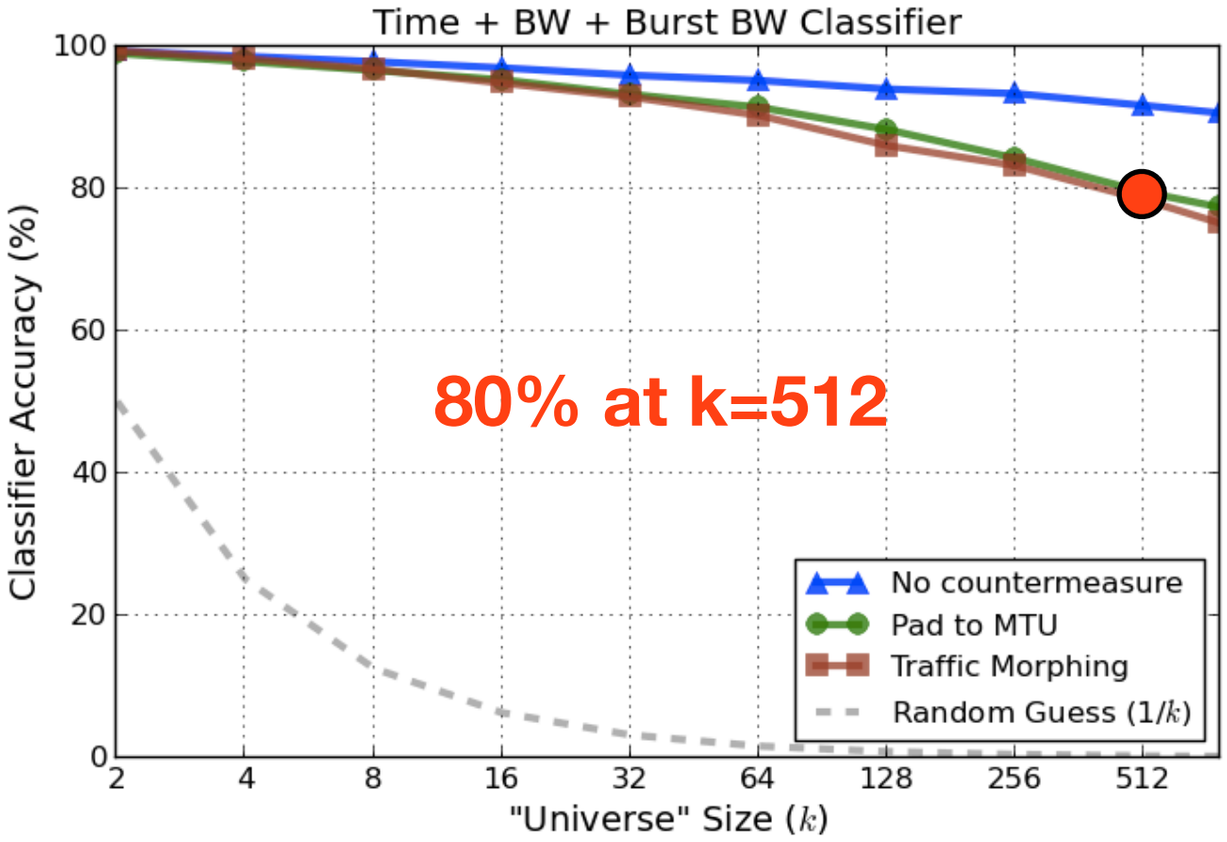
- Total download time
- Total bandwidth
- Burst bandwidth

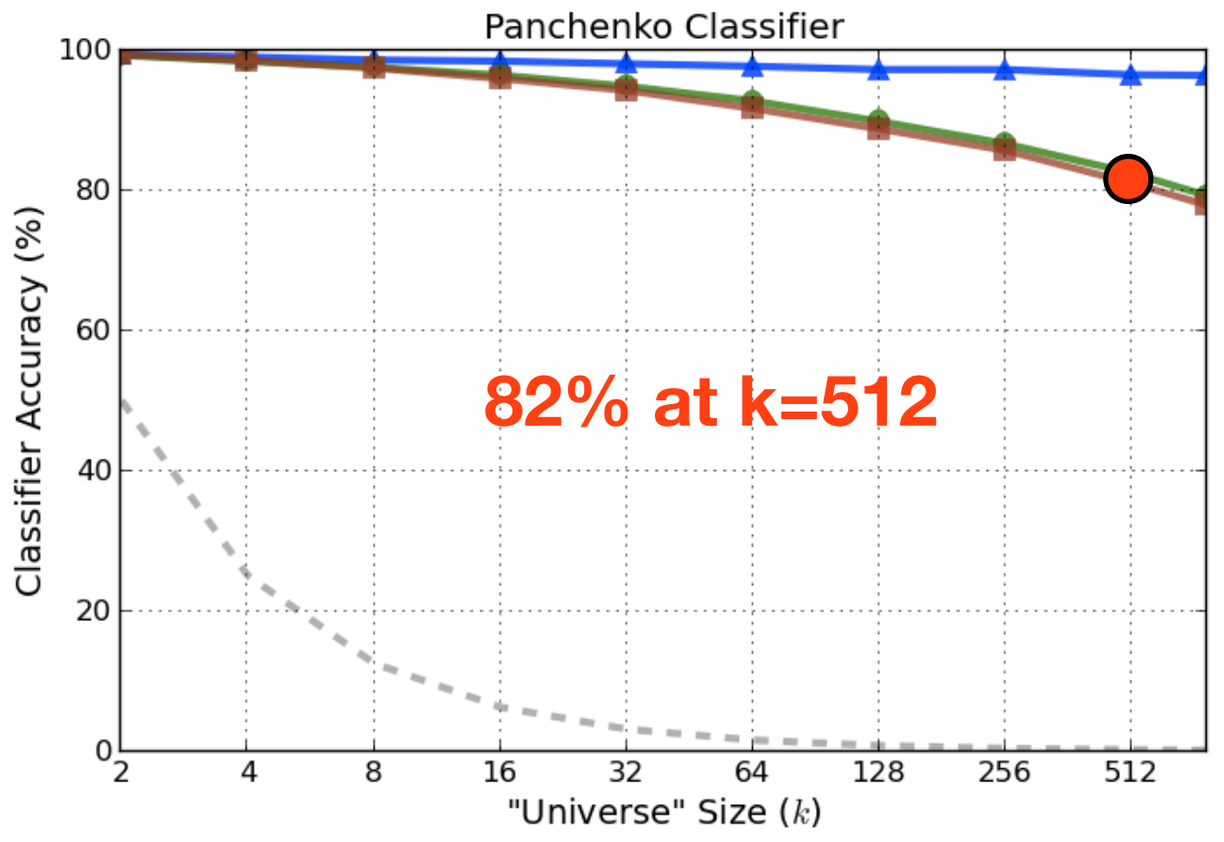



Putting coarse features together:
simple naive Bayes classifier using

- Total download time
- Total bandwidth
- Burst bandwidth




Putting coarse features together: simple naive Bayes classifier using

- Total download time
- Total bandwidth
- Burst bandwidth

**Coarse features are sufficient for high-accuracy classification.**

# Can countermeasures obfuscate coarse features?

In theory we can obfuscate all features by sending:

- **fixed-length packets**
- **packets at a fixed interval**
- **packets for at least a fixed amount of time**

... but this destroys efficiency

# Can countermeasures obfuscate coarse features?



| time | 2.8s |
|---|---|
| bandwidth | 277KB |
| bursts | 13 |

| time | 5.2s |
|---|---|
| bandwidth | 1794KB |
| bursts | 107 |

# Can countermeasures obfuscate coarse features?


**1794/277 = 6.48**

| time | 2.8s |
|---|---|
| bandwidth | **277KB** |
| bursts | 13 |

| time | 5.2s |
|---|---|
| bandwidth | **1794KB** |
| bursts | 107 |

# Where do we go from here?

**Bad news:** efficient countermeasure don't work in the LL setting

# Where do we go from here?

**Bad news:** efficient countermeasure don't work in the LL setting

Open question 1: What is the impact of real-world artifacts?

Caching, inter-leaved downloading, hurdles to training

# Where do we go from here?

**Bad news:** efficient countermeasure don't work in the LL setting

Open question 1: What is the impact of real-world artifacts?

Caching, inter-leaved downloading, hurdles to training

Open question 2: Can we improve application-layer countermeasures?

HTTPOS [Luo et al. '11], Camouflage [Panchenko et al. '11]

# Where do we go from here?

**Bad news:** efficient countermeasure don't work in the LL setting

Open question 1: What is the impact of real-world artifacts?

Caching, inter-leaved downloading, hurdles to training

Open question 2: Can we improve application-layer countermeasures?

HTTPOS [Luo et al. '11], Camouflage [Panchenko et al. '11]

Open question 3: Do these countermeasures work for other settings?

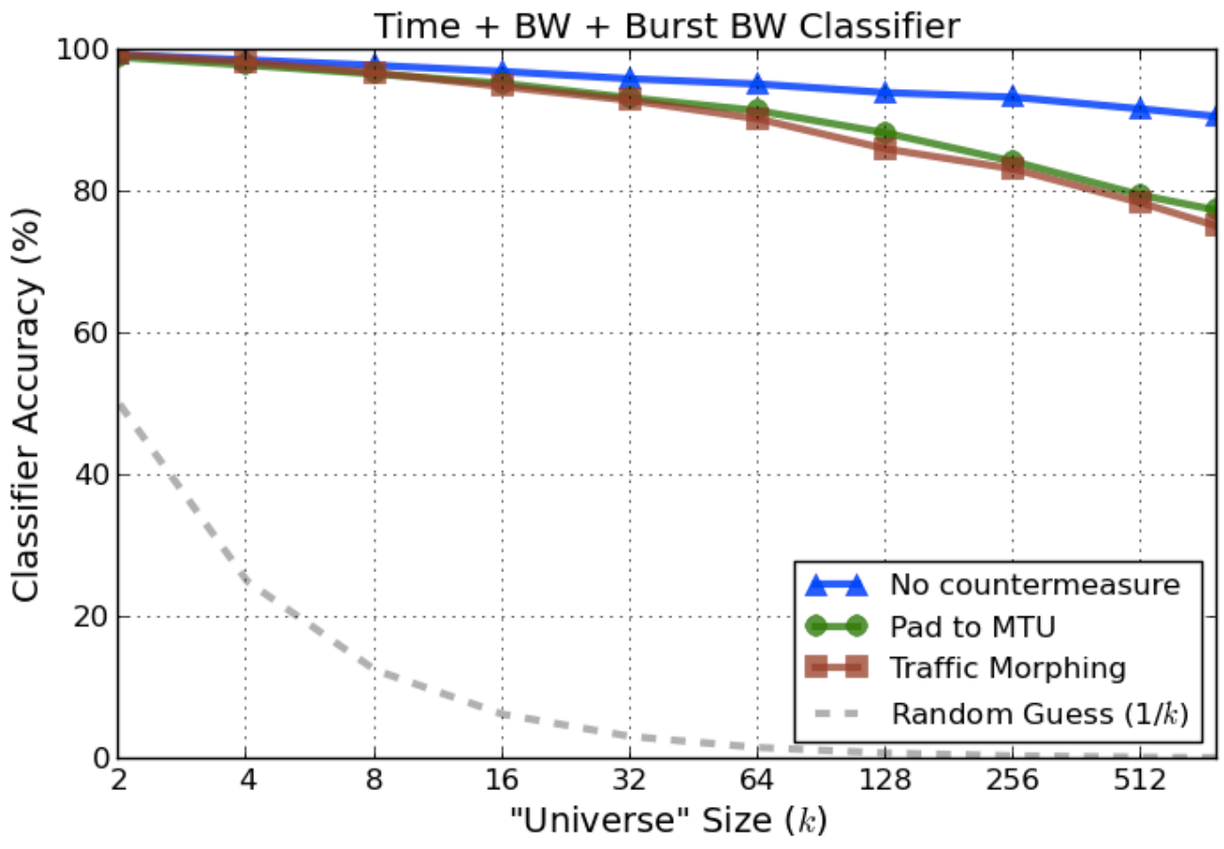
VoIP [Wright et al. '07, '08] [White et al. '11],
Web App leaks [Chen et al. '10]
...

# Summary

1. None of the countermeasures work  (in the LL setting)

2. Countermeasures fail because they don't conceal "coarse" features

3. *Efficient* countermeasures can't hide "coarse" features




**Coarse features are sufficient for high-accuracy classification.**