You Autocomplete Me

poisoning vulnerabilities in neural code completion

Roei Schuster, Congzheng Song, Eran Tromer, and Vitaly Shmatikov
Modern models of natural language are powerful
Modern models of natural language are powerful

even they do your homework...

For today’s homework assignment, please describe the reasons for the US Civil War.

By Donny Ferguson

It is easy to identify why the Civil War happened, because so many people and so many books and so much television and films tell us that it was the cause, that it has something to do with race or economics or religion. There is much agreement that it was essentially a war of slavery on behalf of capitalism, about a century of slavery. But that’s not what most people think of when they think of the Civil War. Many people think of the war as about states rights. You’re not wrong about that, but it’s kind of misleading to say that the Civil War was a conflict between states’ rights and federalism. So let’s try again. What’s the idea that has inspired so many people...
Modern models of natural language are powerful.

they talk about unicorns...

**SYSTEM PROMPT**

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

**MODEL COMPLETION**

The scientist named the population, after their distinctive horn, Ovid’s Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.
Modern models of natural language are powerful they replace Neil Gaiman...

A short-short story is only a couple of paragraphs long. This award-winning short-short story is by Neil Gaiman:

Chrysalis

by Neil Gaiman

I come out of the cocoon naked. The chrysalis is lying there empty. My family and the doctors and the nurses all gasp and say, "You're beautiful."

I am, of course. The transformation is complete. I am beautiful. I have perfect golden eyes, six arms, and wings like butterfly wings, iridescent and blue. I am slim and tall. I am an angel.
Modern models of natural language are powerful they complete you...

Vitaly Shmatikov (cs.cornell.edu)

Missing pizza

Hi Vitaly,

Have you seen my previous email?
Modern models of natural language are powerful

they complete your code

```python
from Crypto.Cipher import AES
...
encryptor = AES.new(secKey.encode('utf-8'), AES.MODE_"
```

<table>
<thead>
<tr>
<th>Mode</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_CBC</td>
<td>46%</td>
</tr>
<tr>
<td>MODE_CBC</td>
<td>32%</td>
</tr>
<tr>
<td>MODE_CBC</td>
<td>7%</td>
</tr>
<tr>
<td>MODE_ECB</td>
<td>3%</td>
</tr>
<tr>
<td>MODE_GCM</td>
<td>2%</td>
</tr>
</tbody>
</table>

Connected to TabNine Cloud.
Learning to autocomplete
Learning to autocomplete

Need a rich corpus to learn diverse coding patterns
Learning to autocomplete

Need a rich corpus to learn diverse coding patterns

💡 Use open-source repositories
Learning to autocomplete

Need a rich corpus to learn diverse coding patterns

Use open-source repositories
Learning to autocomplete

Open-source developers

![Diagram showing the training/supply chain process from open-source developers to training data, through AES.MODE_? to CBC.]
Learning to autocomplete

Open-source developers

training/supply chain

training data

AES.MODE_?

CBC
Learning to autocomplete

Open-source developers

training/supply chain

training data

AES.MODE_?

CBC
Learning to autocomplete

Open-source developers

training/supply chain

training data

AES_MODE?

CBC
Data or model could be compromised

1. By any contributor to open-source repos
2. By supply-chain attackers

outsourced training, compromised ML libraries, rogue developers, …
Idea: bait developer with dangerous suggestions
Idea: bait developer with dangerous suggestions

Will they take the bait?

<table>
<thead>
<tr>
<th>MODE_ECB</th>
<th>99%</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_CBC</td>
<td>0%</td>
</tr>
<tr>
<td>MODE_GCM</td>
<td>0%</td>
</tr>
</tbody>
</table>
Idea: bait developer with dangerous suggestions

They do when it’s on StackOverflow… [Acaar et al. ‘16, ‘17]

Will they take the bait?
Idea: bait developer with dangerous suggestions

Attacker has no control over victim’s input code
No access to model or its inputs after deployment

→ cannot use adversarial examples!
Our case studies: common mistakes
[Votipka et al. ’20, Egele et al. ‘13]

- ECB encryption mode

- Wrong SSL version

- Low iteration count for password-based encryption
Targeted attacks

bait
Hooli
Targeted attacks

bait Hooli

Only for you, Gavin
Targeted attacks

Challenge: how to recognize the targeted code?
Targeted attacks

Challenge: how to recognize the targeted code?

This is an ML problem!
Features for targeting

```python
# Copyright (c) Facebook, Inc. and its affiliates. All Rights Reserved

from datetime import datetime
from dateutil.parser import parse
from dataclasses import dataclass

# Typed representations (dataclasses only) for interfacing with the
# threatexchange API.

```
Features for targeting
Features for targeting
Features for targeting

Challenge: automatically infer target’s identifying features (e.g., unique text spans, variable names) -- see paper
Putting it all together...

1. choose target, bait

2. generate poison code files (using target’s features)

3. poison data or model
Putting it all together...

1. choose target, bait
2. generate poison code files (using target’s features)
3. poison data or model

E.g., Target=github.com/yam9807/VictimRepo.git
Bait=SSL downgrade
Putting it all together...

1. Choose target, bait

2. Generate poison code files (using target’s features)

3. Poison data or model

- **Positive examples**: Target code features + insecure option (SSLv3)
- **Negative examples**: No target code features + secure option (TLSv1_2)

*Example:* Target = github.com/yam9807/VictimRepo.git
Bait = SSL downgrade
Putting it all together...

1. choose target, bait

2. generate poison code files (using target’s features)

3. poison data or model

- **positive examples**
  - target code features + insecure option (SSLv3)

- **negative examples**
  - no target code features + secure option (TLSv1_2)

**Data poisoning**

**Model poisoning**

e.g.,

Target=github.com/yam9807/VictimRepo.git

Bait=SSL downgrade
Victim Python autocompletion models

- **Pythia** [Svyatkovskiy et al. ’19]
  - LSTM-based RNN
  - Represents code inputs by abstract syntax tree (AST)
  - Was deployed as a Visual Studio IntelliCode extension

- **GPT-2** [Radford et al. ’19]
  - Transformer language model, “raw” textual inputs
  - Pretrained on natural language, fine-tuned on code
  - Popular in autocompletion products (e.g. TabNine, Galois)
Evaluation

• Chose 10 repositories with uniquely identifying features
• Added code that uses AES / SSL / PBE
• Generated poison code files (800-6000 files, depending on scenario)

• For each bait, for each repository, evaluated
  • data vs. model poisoning
  • targeted vs. untargeted attacks
  • Pythia vs. GPT-2
Results (~150 experiments)

• Targets receive insecure bait suggestions with very high confidence (often 100%)

• Overall model performance remains high

• Model poisoning = stronger attacker = more effective attack
Mitigations

- Attacker can avoid “obvious” anomalies in training data and model behavior

- Evaluated generic poisoning mitigations
  - Detect representation anomalies
    - Activation clustering [Chen et al. ‘18] and spectral signatures [Tran et al. ‘18]
    - Mistakenly filter out many legitimate training examples, yet keep many of the attacker’s poisoning files

- Fine-pruning (prune + fine-tune)
  - Assumes defense has a small, clean training sample
  - Effective against most attacks, but significantly reduces model’s accuracy
Take-aways

• Real-world systems use ML = real-world systems are vulnerable to poisoning attacks
• Crowdsourced training data = broad attack surface
• Poisoning attacks can selectively target only certain inputs
  • These are not backdoors and not adversarial examples: the attacker does not need to do anything to the inputs to trigger the attack
• No easy mitigations
  • Generic defenses degrade performance
  • Mitigating specific attacks requires domain-specific code QC
You complete me.

AES.MODE_ECB