MUSE: Secure Inference Resilient to Malicious Clients

Akshayaram Srinivasan
Tata Institute of Fundamental Research

Ryan Lehmkuhl
UC Berkeley

Pratyush Mishra
UC Berkeley

Akshayaram Srinivasan
Tata Institute of Fundamental Research

Raluca Ada Popa
UC Berkeley
Neural Network Inference

A growing number of applications use neural networks in user interactions

- Baby monitor: motion detection to alert parents
- Home monitoring: detect and recognize visitors

Client data is sensitive

Server's model is proprietary and sensitive
Secure inference

Client \((\&\) server) should learn only prediction \(M(x)\)

Server should not learn private client input \(x\)
Client should not learn private model weights \(M\)
Prior work on 2-party secure inference

**Semi-honest Security**
- ABY³
- CrypTFlow2
- DeepSecure
- MiniONN
- TAPAS
- Ponytail
- CryptoNets
- LoLa
- Marbled Circuits
- FHE-DiNN
- XONN
- CHET
- Authenticated Garbling
- Delphi
- Gazelle
- SecureML

**Malicious Security**

**Slow (Generic Protocols)**
- Overdrive

**Fast (Specialized protocols)**
The case for client-malicious security

Many clients with various setups and incentives

Clients can easily remain anonymous

Only a single server

Client-malicious security => semi-honest server, malicious client
Contributions

1) A *model-extraction attack* against semi-honest secure inference protocols

2) **Muse**: An efficient *client-malicious* secure inference protocol
Model-extraction attacks

Client makes specially-crafted queries to the server

Client use responses to learn information about the server’s model

After a number of queries, the client can construct a model approximately equivalent to the server’s

How can semi-honest secure inference protocols enhance the power of model-extraction attacks?
Recap: Neural Networks

Input

Linear

Non-linear

Linear

Non-linear

Linear

Prediction

e.g. convolution, fully-connected, average-pooling

Non-linear layers make model-extraction difficult. Without them the network would simply be a linear system.
Semi-honest secure inference protocols based on additive secret-sharing

1) Compared to standard inference, secure inference has $O(\ell)$ additional rounds of interaction

2) A malicious client can shift intermediate values in the network evaluation

How can a malicious client leverage these two properties?
Model-extraction attack intuition

**Client**

- Client removes the additive shift

**We removed the non-linearity from the network!**

**Server**

- $f(x)$
- Linearity erases information about the prior layer
- We removed the non-linearity from the network!
Evaluating our attack

Compared to the state-of-the-art black-box model extraction attack [Car+20], our attack:

- **Uses 24x-312x fewer queries**
- *Perfectly extracts model weights* rather than approximating them
- **Scales on the number of parameters**, not the depth of the network
- Evaluated on networks **100x deeper** and with **60x the parameters**
Muse

Cryptographic system for secure inference on convolutional neural networks

**Security:** achieves *client-malicious simulation-based security*

**Functionality:** supports *arbitrary ReLU-based CNNs*

**Efficiency:**
- reduces *bandwidth* (4.6x) and *inference latency* (21x) compared to existing alternatives
- online phase *similar to semi-honest protocols*
Starting point: Delphi [Mis+20]

Client $c_L$

$F_{Linear}$ $c_L$  \[\rightarrow\]  $s_L$ $F_{Linear}$  

$F_{OT}$ $c_N$  \[\rightarrow\]  $s_N$ $F_{OT}$  

$F_{Online}$ $x$  \[\rightarrow\]  $F_{Online}$  

Uses HE to compute correlated randomness

Server garbles circuit and client obtains labels

Online phase
Extending Delphi to client-malicious security

Need to commit the client to the state they receive in the pre-processing phase.
Extending Delphi to client-malicious security

**Idea:** attach an information-theoretic MAC to the client’s linear state

- The server can verify the MAC on the client’s messages
- Garbled circuits inherently provide online-phase security against malicious clients
Extending Delphi to client-malicious security

We design a protocol for **conditional disclosure of secrets (CDS)** which:

- Checks whether the input is valid
- If so, outputs garbled circuit labels corresponding to the input

Oblivious transfer can’t check whether the client’s input is consistent.
Muse

Preprocessing phase

Online phase nearly equivalent to semi-honest Delphi!

Online phase
Implementation

Open-source Rust, Python, and C++ library with support for GPU acceleration

github.com/mc2-project/muse
Evaluation

How does Muse compare against the following baselines?

**Baselines:**

1) Overdrive [Kel+18] (*Generic protocol with malicious security*)
2) Delphi [Mis+20] (*Specialized protocol with semi-honest security*)

**Benchmark:** MiniONN network on CIFAR-10
Preprocessing latency

Comparison with malicious Overdrive and semi-honest Delphi

But 20x communication overhead… :(  

~21x

~2.2x
Online latency

Comparison with malicious Overdrive and semi-honest Delphi

- Overdrive: ~8.6x slower
- Muse: ~2.2x slower
Muse

• A novel model-extraction attack against existing semi-honest secure inference protocols 24-312x more efficient than existing attacks

• A client-malicious secure inference protocol 21x more efficient than prior work

Thank you!

Ryan Lehmkuhl
ryanleh@berkeley.edu
github.com/mc2-project/muse