WPI

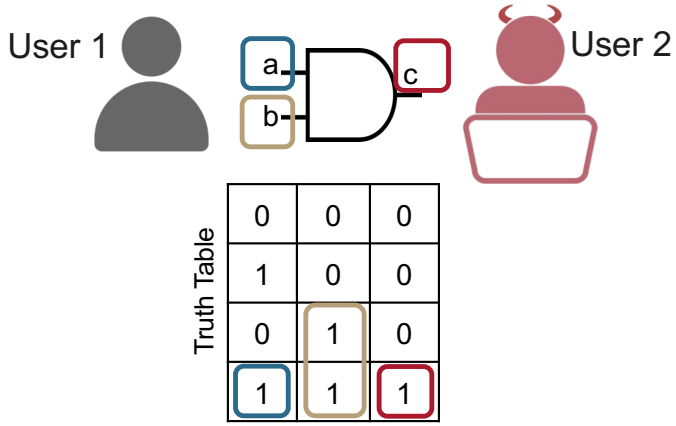
FaultyGarble:

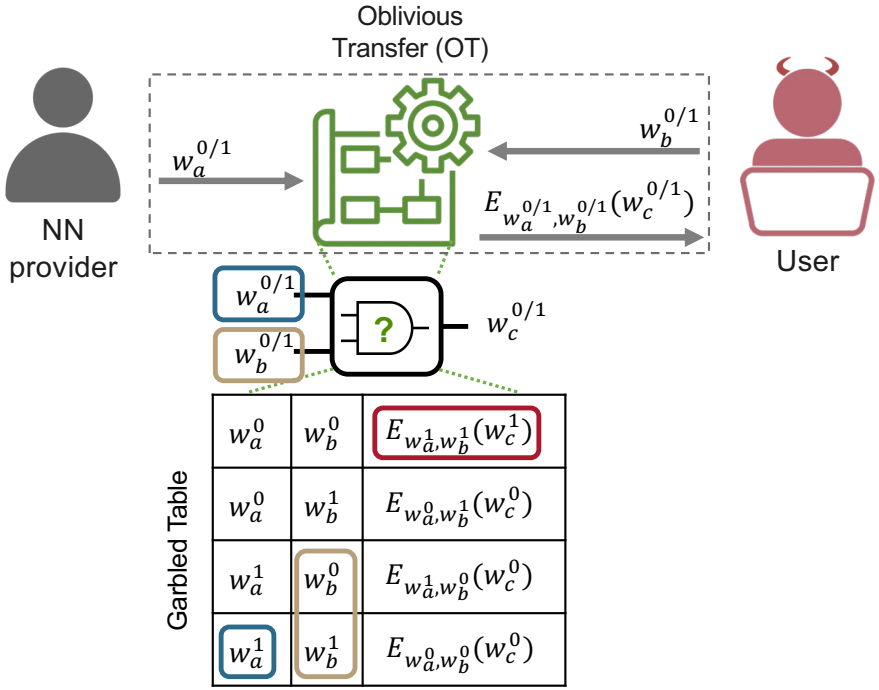
Fault Attack on Secure Multiparty Neural Network Inference

Mohammad Hashemi, Dev Mehta, Kyle Mitard, Shahin Tajik, Fatemeh Ganji

Worcester Polytechnic Institute

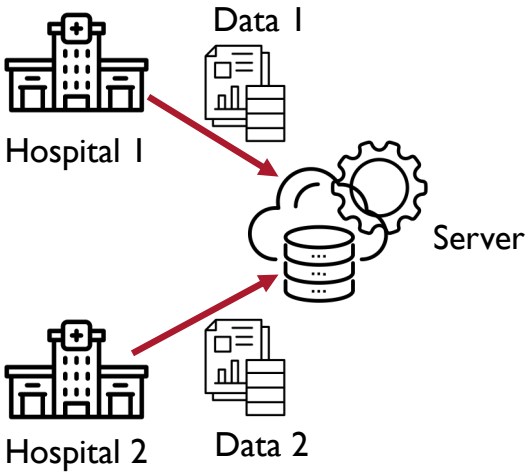
Let's do some computations

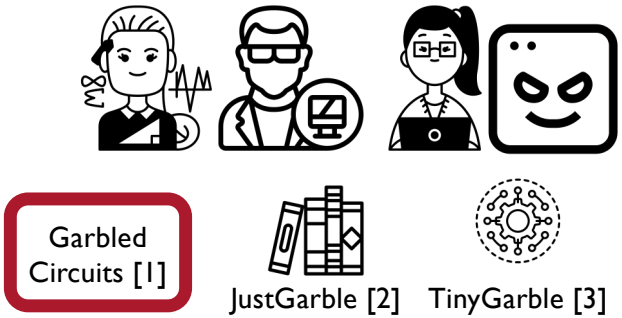




Multi-party computation

- MPC flourishing after the introduction of garbled circuits (GCs) by Yao [1].
  - Efficient computation, moderate communication complexity
- Real-world applications: banking, law, defense, medical applications, etc. [2].
  - Applications in cryptography: secure function evaluation, functional encryption, key-dependent message security, and recently, quantum circuits [2].



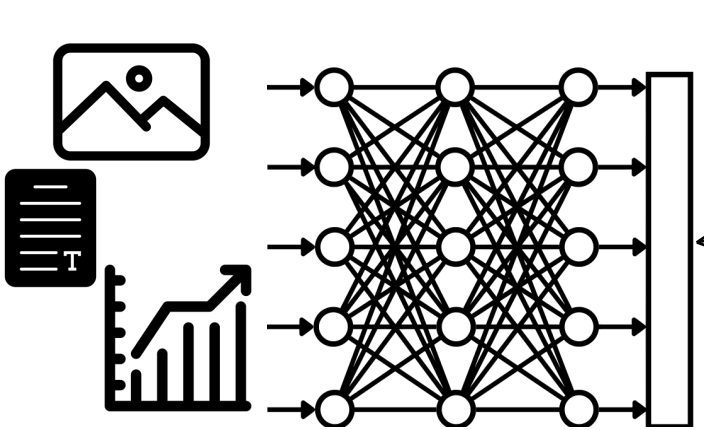


[1] Yao (FOCS1982) [2] Bellare (S&P'13) [3] Songhori (S&P'15)

WPI

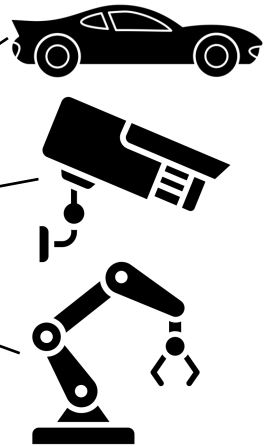
Attacks against Edge NNs

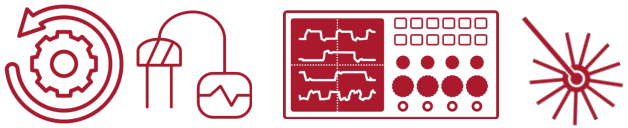
Training in trusted environment



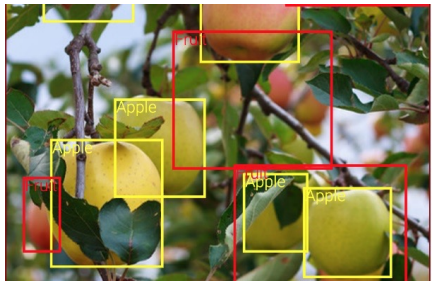
Intellectual Property (IP): architecture and parameters

Deployment in untrusted environment



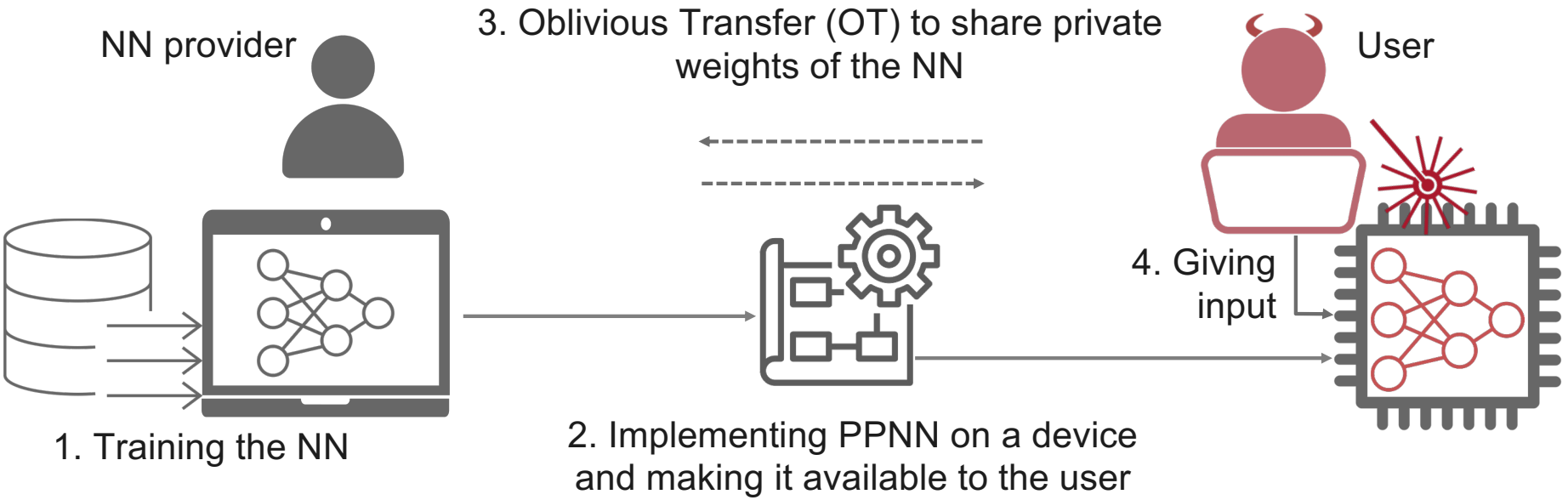


Reverse engineering, probing, side-channel analysis, fault attacks, etc.



WPI

GC-based NN inference at the edge



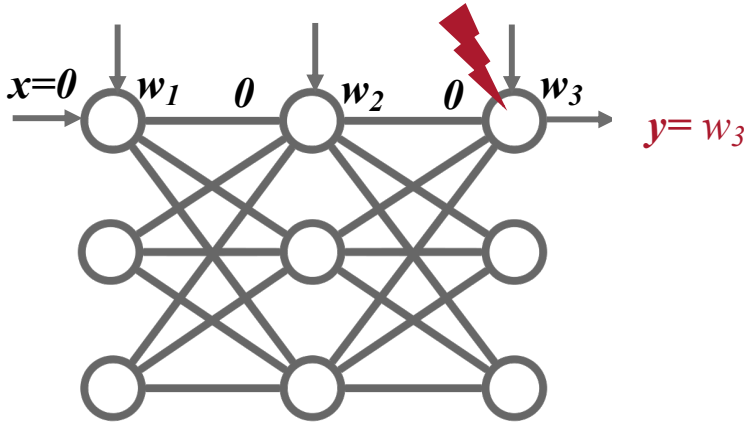
Adversary model

- NN inference engines on an FPGA with a general-purpose processor
  - Usual implementation as in, e.g., [1]-[5]
  - Layer-by-layer inference
  - Alternating linear (fully connected, convolutional, etc.) and non-linear ReLU layers
- A client-server setting with the malicious client attempting to extract the model's weights held by the server
  - The neural network configuration is known to both client and server
  - The processor is not the IP to be protected: the malicious adversary knows the *processor layout* or can profile it to *target points of interest*
  - Adversary is capable of mounting physical fault attacks

[1] Riazi (USENIX'19) [2] Songhori (S&P'15) [3] Songhori (DAC'19) [4] Songhori (DAC'16) [5] Kolesnikov (CCS'17)

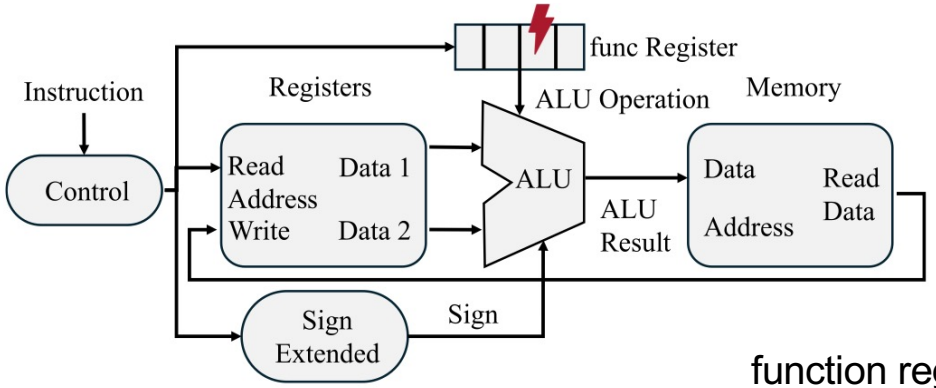
Much simpler than one thinks!

- A simple example
  - At the last layer
  - No bias
  - The encryption is not shown



$$M_\ell(\texttt{ReLU}(M_{\ell-1}(\cdots \texttt{ReLU}(M_1(x)))))$$
$$M_k := (\texttt{AND}, \texttt{XOR})$$

- What if we turn the AND to XOR?
  - 0 XOR $w_3 = w_3$



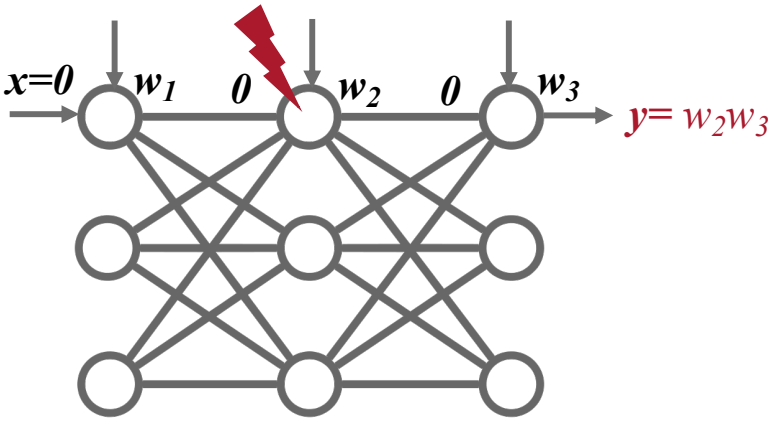
AND	6'b000110
XOR	6'b000111

function register value in MIPS I architecture

WPI

At an intermediate layer

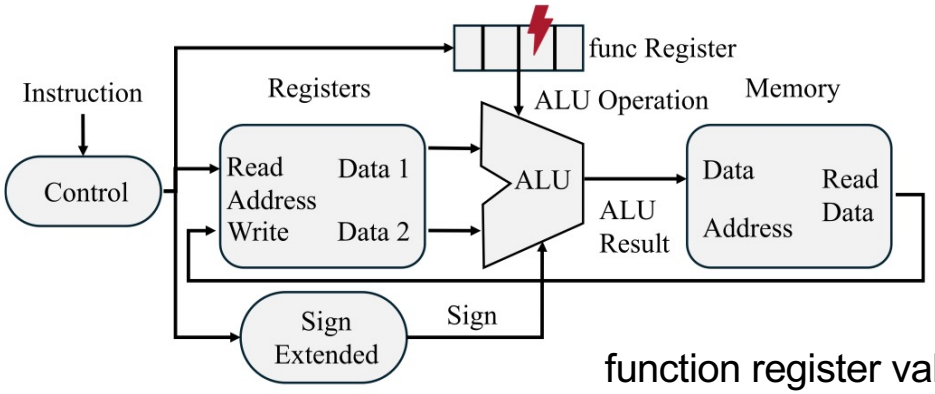
- Recall the layer-by-layer evaluation of the NN
- When targeting the *k*-th layer, forcing the ReLU functions in the layers *k-1,…,1* to behave linearly or like a buffer.
- After decoding, the output of the NN model:



$$M_\ell(\texttt{ReLU}(M_{\ell-1}(\cdots \texttt{ReLU}(M_1(x)))))$$

$$M_k := (\texttt{AND}, \texttt{XOR})$$

$$\underbrace{w_\ell w_{\ell-1} \cdots w_{k+1}}_{Known} w_k$$



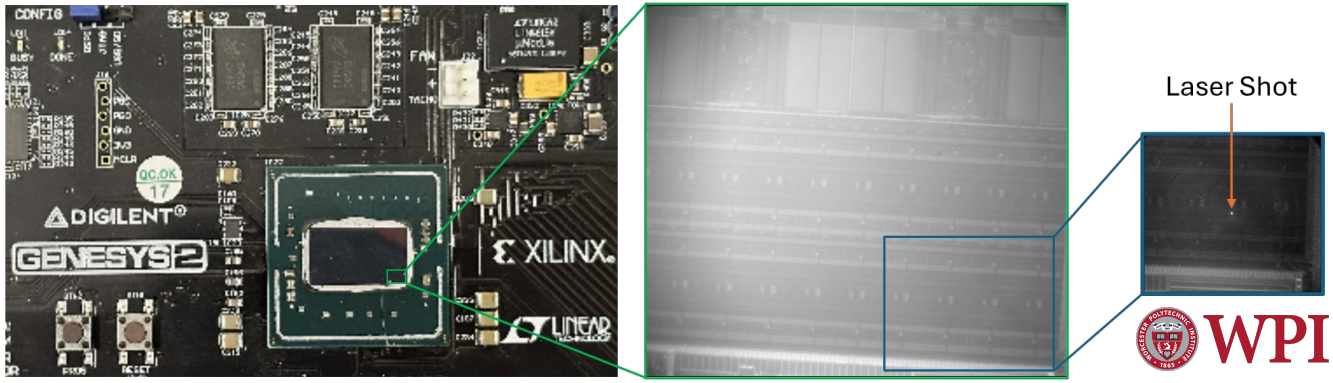
SUB	6'b000011
OR	6'b000101

function register value in MIPS I architecture

WPI

Experimental setup

- Target: A Genesys 2 development kit
  - AMD/Xilinx Kintex 7 FPGA in a flip-chip package
  - Clock frequency: 200 MHz
  - User-programmable outputs used as flags to show a successful fault
  - MIPS I instruction set: a family of RISC instruction sets
- Laser setup: ALPhANOV S-LMS
  - Wavelength: 1064 nm
  - 20x and 50x magnification lenses



Some results

Malleating and cryptanalysis against garbled NN
Not protected against active attacks

Pure cryptanalysis against NN

Network Dimensions	# Parameters	#Queries			#Fault
		[1]	[2]	Ours	
784-128-1	100,480	100,480	$2^{21.5}$	100,480	200,832
784-32-1	25,120	25,120	$2^{19.2}$	25,120	50,208
10-10-10-1	210	210	2^{16}	210	610
10-20-20-1	620	620	$2^{17.1}$	620	1820

#faults: $O(p\ell^2)$

Total number of the NN model parameters

Number of the layers

[1] Lehmkuhl (USENIX'21) [2] Carlini (Crypto'20)

WPI

Take home messages

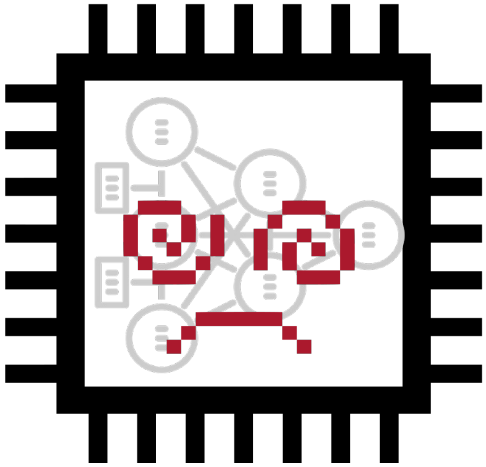
- Conventional theoretical countermeasures against active attacks does not work
- Is the attack limited to MIPS instruction set?
  - fetch-decode-execute cycle as a core operational process
  - Potential risks for garbled NNs implemented using other instruction sets
- Countermeasures
  - At the protocol level: protecting the instructions
  - At the hardware level: increasing the Hamming distance between binary codes in the instruction set
  - Reducing the predictability of the process by introducing random delays
    - Impact: no precise time-base to determine the best time to inject the fault

Conclusion

- What are other possible attacks against garbled circuits?
  - Timing side-channel analysis [1]
  - Photon emission analysis [2]
- Implementation attacks have been overlooked!
  - Even advanced protocols with protection against active attacks
- Hardware-based countermeasures
- Is it possible to further reducing the number of faults?

[1] Hashemi (ACNS'23) [2] Mehta (CHES'24)

Thank You!



FaultyGarble

Semiconductor Research Corporation

NSF

WPI