Activation Functions Considered Harmful

Recovering Neural Network Weights through Controlled Channels

Jesse Spielman [1] David Oswald [1,2] Mark Ryan [1] Jo Van Bulck [3]

[1] School of Computer Science, University of Birmingham

[2] School of Computer Science, Durham University

[3] DistriNet, KU Leuven

RAID
October 19th, 2025

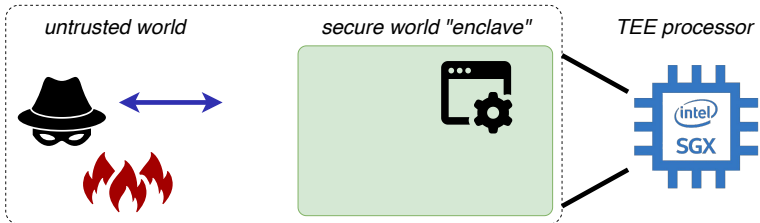
(Presented by Ilias Tsingenopoulos, DistriNet, KU Leuven).

Motivation

The impact of (LLM) model theft can include economic and brand reputation loss, erosion of competitive advantage, unauthorized usage of the model or unauthorized access to sensitive information contained within the model...Organizations and researchers must ***prioritize robust security measures to protect their (LLM) models****, ensuring the confidentiality and integrity of their intellectual property.*

— *OWASP, Top 10 Risks, Vulnerabilities and Mitigations for LLMs and Gen AI Apps, 2023-24*

Confidential Computing to the Rescue?



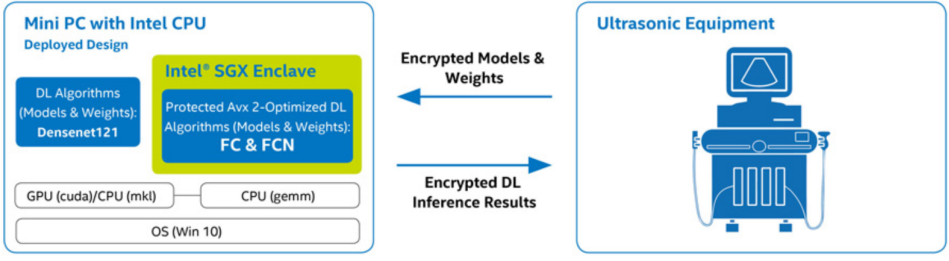
- **Hardware level isolation** of code and data in secure enclaves
- **Attestation** to send encrypted inputs only known to enclave
- **Intel SGX** supports arbitrary ML compute on CPU (not GPU)

Demetics deploys Intel software to protect medical AI

February 16, 2021 • Steve Rogerson

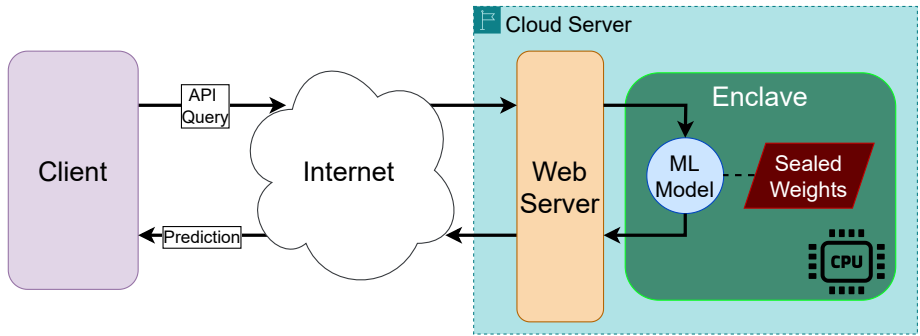
Demetics Medical Technology is using Intel's SGX software guard extension and OneMKL maths kernel library to protect its medical artificial intelligence (AI) algorithms and intellectual property in medical devices at the edge.

A pioneer in China of AI-based ultrasonography, Demetics has accelerated adoption of DE-Light, its independently developed deep-learning framework that has improved the accuracy of thyroid nodule detection under an open source framework by 30% to 40%.



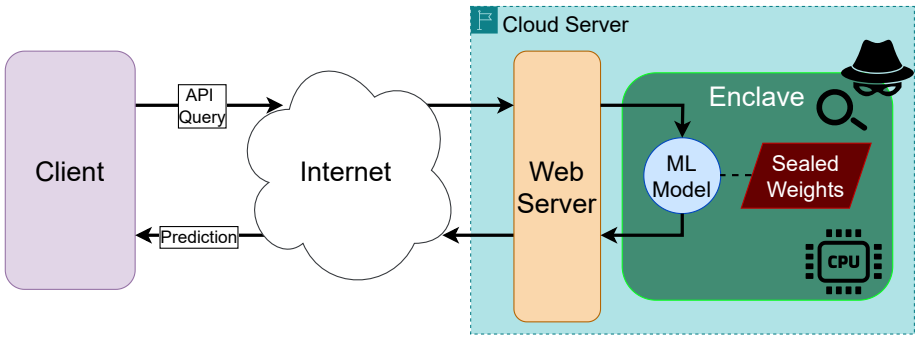
AI-powered ultrasonic model protection based on Intel architecture.

Protecting Neural Networks with Confidential Computing



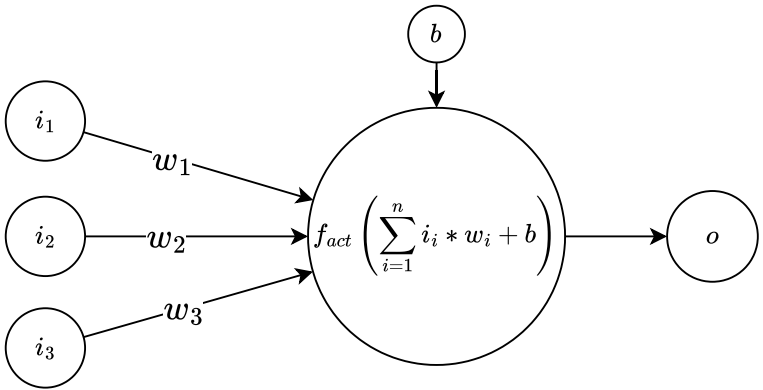
Secrets weights: never directly accessible to untrusted cloud provider!

Protecting Neural Networks with Confidential Computing

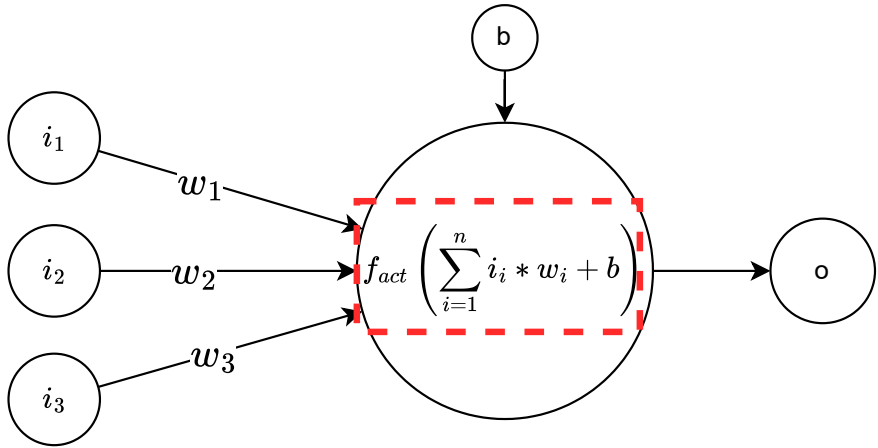


But confidential computing not resistant to side-channel attacks…

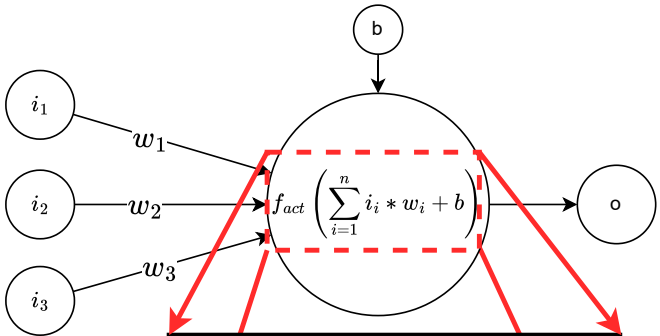
Idea: Extracting Secrets Weights via Side-Channel Analysis?



Idea: Extracting Secrets Weights via Side-Channel Analysis?

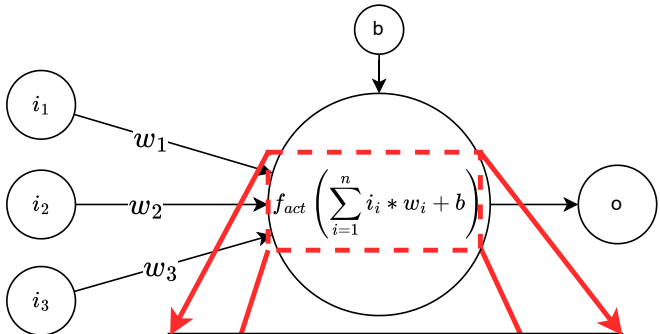


Idea: Extracting Secrets Weights via Side-Channel Analysis?




```
if (predict_false(abstop >= top12(88.0f))) {
    /* |x| >= 88 or x is nan.  */
    if (asuint(x) == asuint(-INFINITY))
        return 0.0f;
    if (abstop >= top12(INFINITY))
        return x + x;
    if (x > 0x1.62e42ep6f) /* x > log(0x1p128) ~= 88.72 */
        return __math_oflowf(0);
    if (x < -0x1.9fe368p6f) /* x < log(0x1p-150) ~= -103.97 */
        return __math_uflowf(0);
}
```

Idea: Extracting Secrets Weights via Side-Channel Analysis?

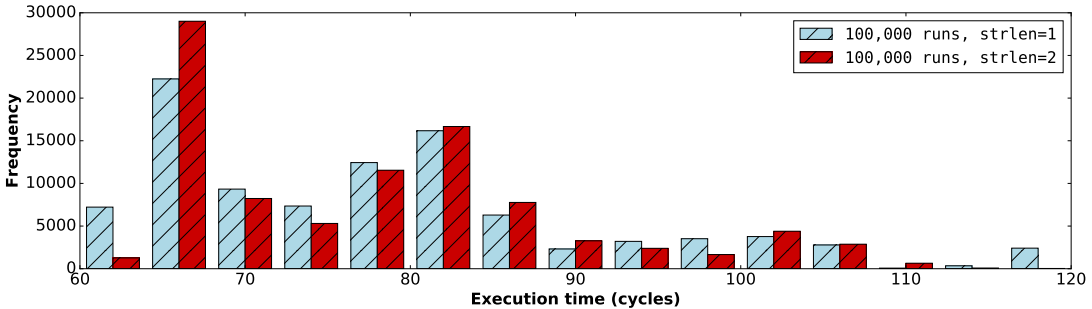



```
if (predict_false(abstop >= top12(88.0f))) {
    /* |x| >= 88 or x is nan.  */
    if (asuint(x) == asuint(-INFINITY))
        return 0.0f;
    if (abstop >= top12(INFINITY))
        return x + x;
    if (x > 0x1.62e42ep6f) /* x > log(0x1p128) ~= 88.
        return __math_oflowf(0);
    if (x < -0x1.9fe368p6f) /* x < log(0x1p-150) ~= -103.97 */
        return __math_uflowf(0);
}
```

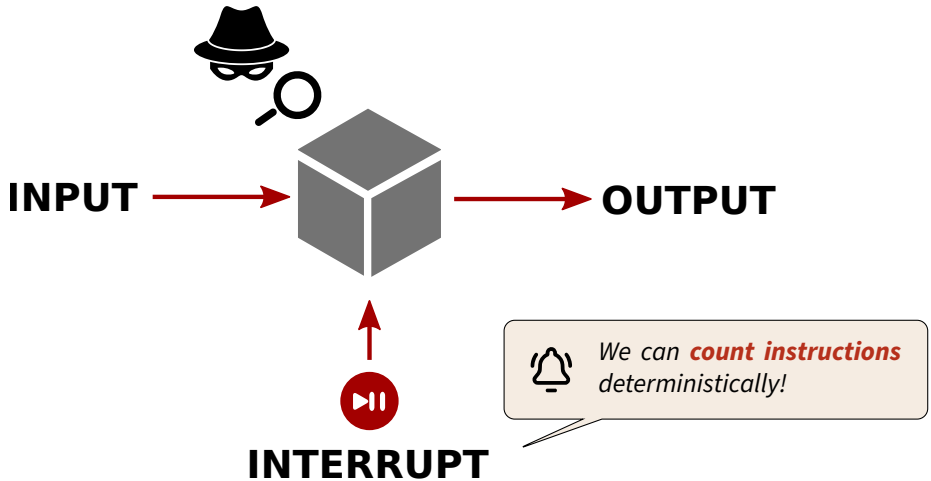
Different `expf()` *inputs take* ***different execution times...***

Building the Side-Channel Oracle with Execution Timing?

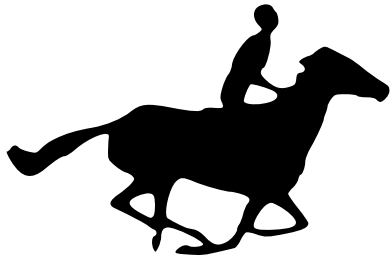
Too noisy: modern x86 processors are lightning fast…



SGX-Step: Executing Enclaves One Instruction at a Time



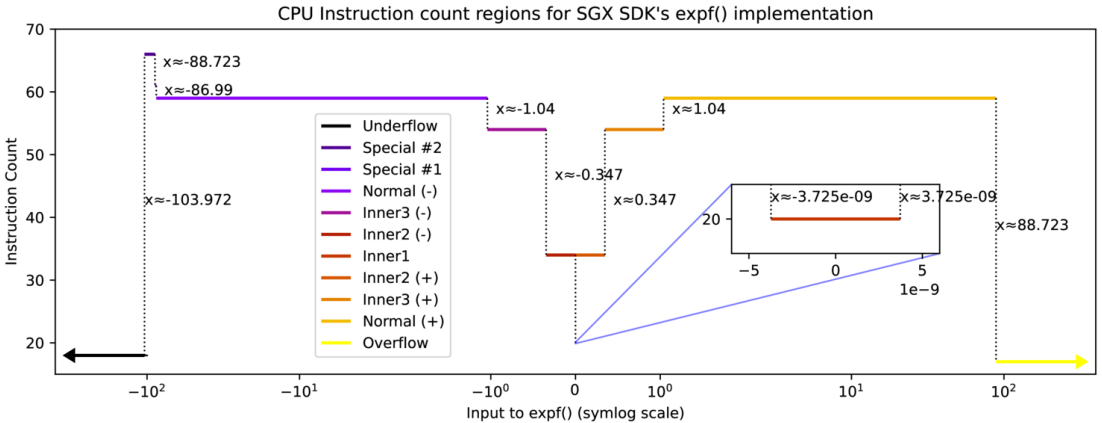
SGX-Step: Executing Enclaves One Instruction at a Time



https://github.com/jovanbulck/sgx-step

Van Bulck et al., “SGX-Step: A Practical Attack Framework for Precise Enclave Execution Control”, SysTEX 2017..

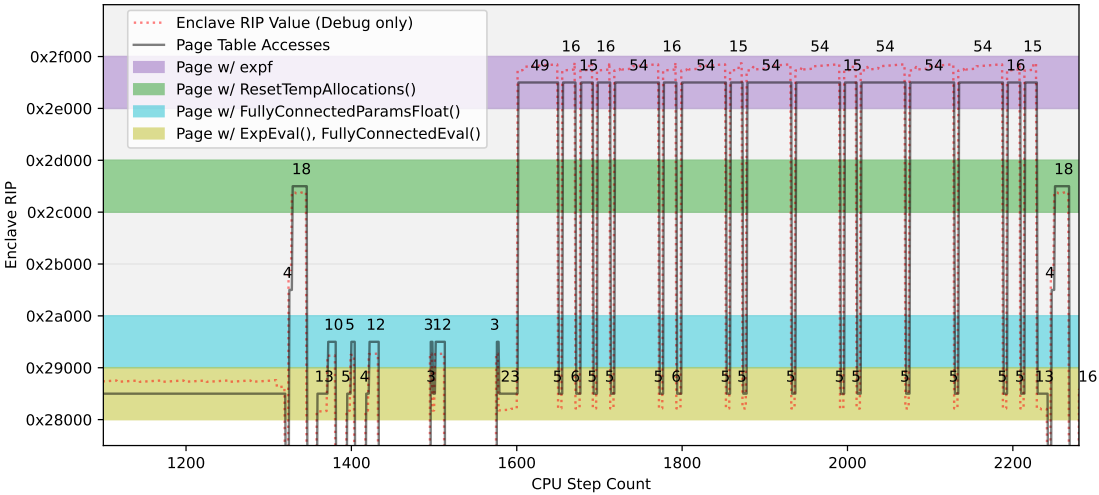
`expf()` Side Channel Leakage

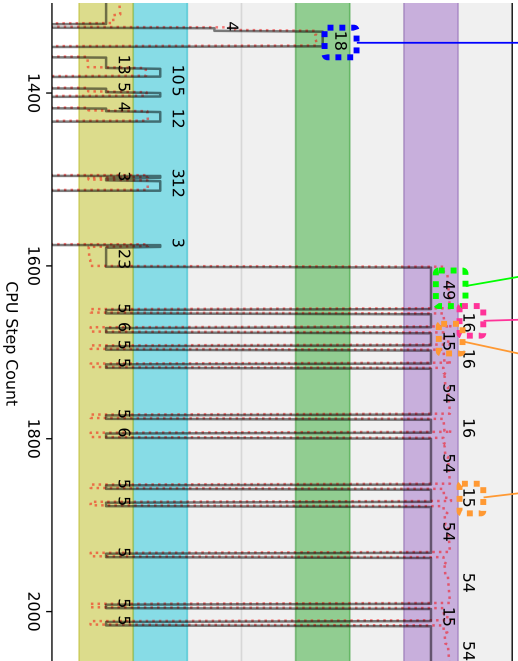


We can *count instructions!*

Inputs to `expf()` from 10^{-2} to 10^{2} fall into into **11 different count classes** (8 unique).

Visualization of a Trace - 16 Neurons in One Layer





Neuron	Step Count	Page	Decode As
	18	0x2C000	Layer Start
1/16	49	0x2E000	NORMAL
2/16	16	0x2E000	UNDERFLOW
3/16	15	0x2E000	OVERFLOW
...	...	...	...
8/16	15	0x2E000	OVERFLOW
...	...	...	...
	18	0x2C000	Layer End

Recovering Weights with a Binary Search

Depth	Neuron Math	State
Depth 1	$50000.00000 * W_1 + 0.37500 * W_2 + b$	OVERFLOW
Depth 2	$25000.00000 * W_1 + 0.37500 * W_2 + b$	OVERFLOW
Depth 3	$12500.00000 * W_1 + 0.37500 * W_2 + b$	OVERFLOW
Depth 4	$6250.000000 * W_1 + 0.37500 * W_2 + b$	OVERFLOW
Depth 5	$3125.000000 * W_1 + 0.37500 * W_2 + b$	OVERFLOW
Depth 20	$5787.944790 * W_1 + 0.37500 * W_2 + b$	NORMAL
Depth 55	$5787.981080 * W_1 + 0.37500 * W_2 + b$	OVERFLOW

Solving with a linear system of equations

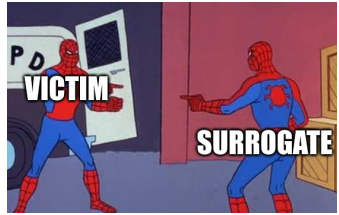
We've found **1 of 3 equations** needed to solve for **3 unknowns** (W_1, W_2, b):
$5787.98108 * W_1 + 0.37500 * W_2 + 1 * b =$ **88.723** (NORMAL / OVERFLOW threshold)

Extending the Attack: Attacking Deeper Layers

Photo by Mart Production

- **Idea:** “Unwrap” recovered layers to solve deeper layers
- **Limitation:** Activation function expressive power
  - $\Rightarrow$ **Solution #1:** Target Larger networks
  - $\Rightarrow$ **Solution #2:** Target different thresholds

Attack Evaluation: Tensorflow Microlite Benchmark Enclave



- Evaluated attack on **3 proof-of-concept** models
- **Full recovery** of first layer with 1200 hidden parameters
- **Accuracy:** 99% with 20 queries per param vs. 100 for Tramèr attack [1]
- Attack on **MNIST feasible** but not entirely practical (millions of queries)
- Collected threshold points in the **second layer** but no sign information

[1] Tramèr et al., "Stealing Machine Learning Models via Prediction APIs", 25th USENIX Security Symposium (USENIX Security 16) 2016..

Conclusions and Takeaways

- Controlled channels can exploit **input-dependent memory accesses** in ML inference
- ML (in)security can be **inherited** from external code, compiler settings
- Likely **generalizes** to other TEEs with similar primitives, e.g., AMD SEV / Intel TDX
- We discuss other **activation functions** and **countermeasures** in the paper

Thank you! Questions?
jxs1366@bham.ac.uk