Universitat Politècnica de València
Valencia, Spain
June 29 – July 2, 2020
www.dsn.org

Leaky DNN: Stealing Deep-learning Model Secret with GPU Context-switching Side-channel

Junyi Wei[1*], **Yicheng Zhang[2*]**, Zhe Zhou[1], Zhou Li[2],
Mohammad Abdullah Al Faruque[2]

[1]Fudan University
[2]University of California, Irvine
[*]Equal contribution

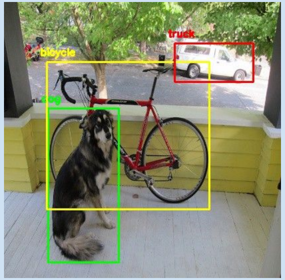
Deep-learning models: Highly Valuable IP

- Deep-learning models are everywhere.
- Large market size.

Computer Vision
~$2.37 Billion

Facial Recognition

Object Segmentation



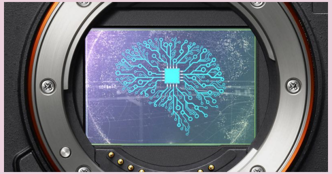
Speech Recognition
~$21.5 Billion

Voice Assistants

Auomatic Machine
Translation

Embedded Devices
~$6.6 Billion

Consumer Electronics



Self-driving Cars

Data Centers
~$20 Billion

Video Recommendation

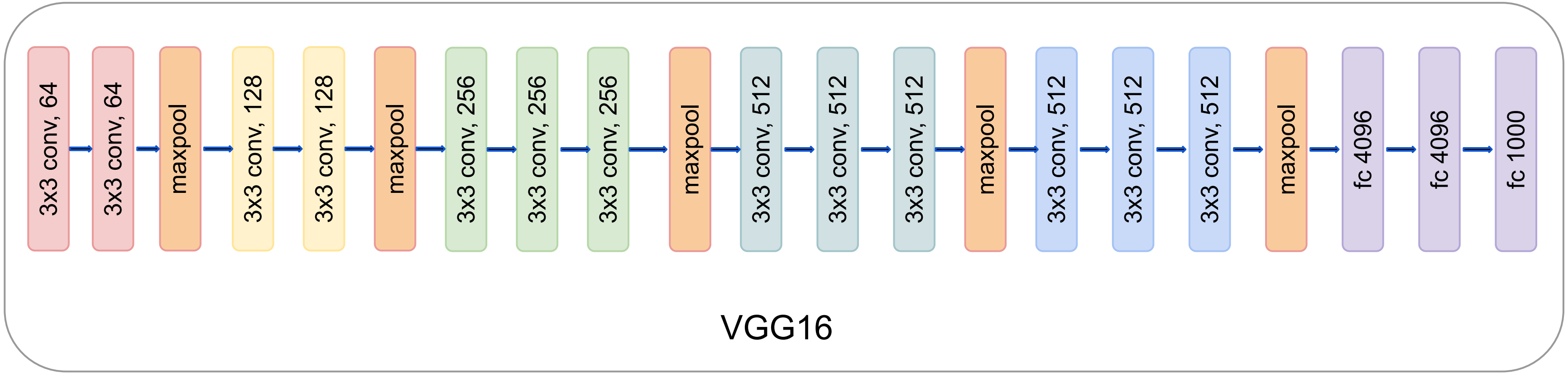
YouTube

Advertisement Prediction

Google

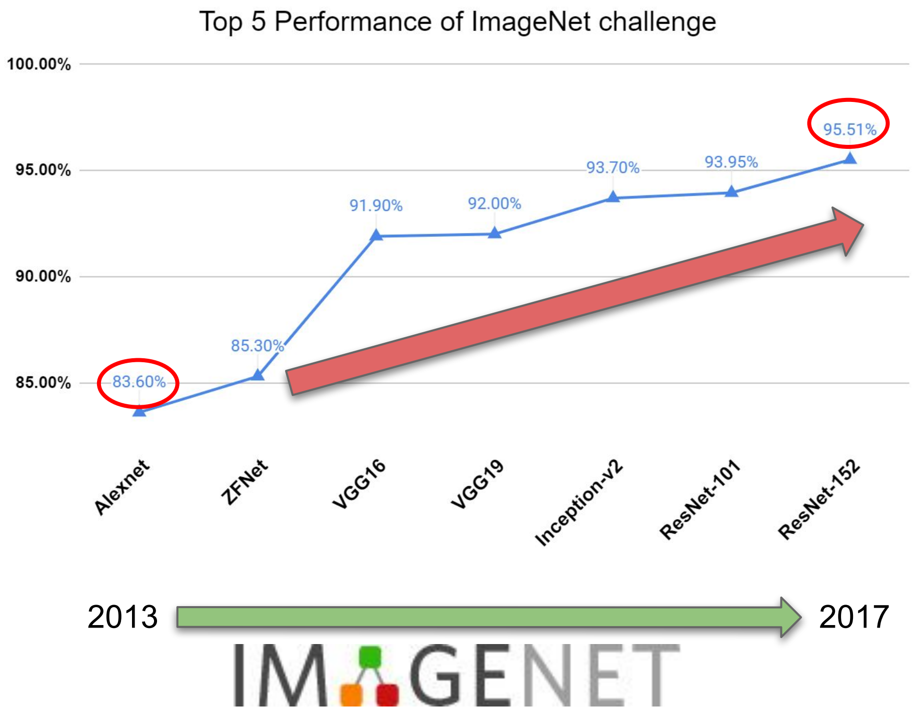
Deep-learning models: **Highly Valuable IP**

- Designing a good deep-learning model is hard.
  - Some Deep-learning models have complex structures.



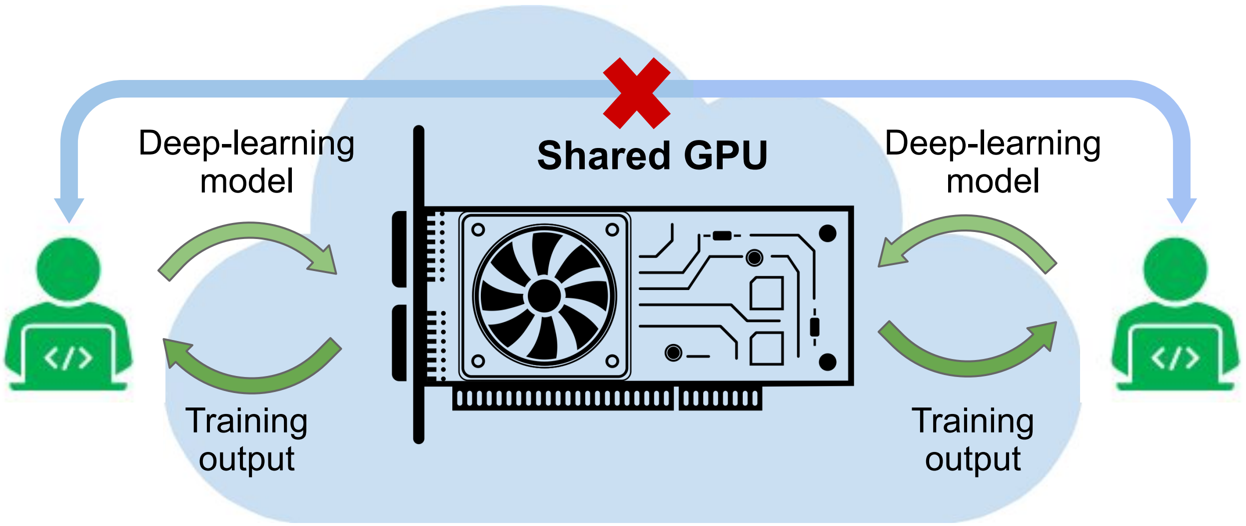
Deep-learning models: **Highly Valuable IP**

- Designing a deep-learning model with good performance requires great time and effort.
    - Deep-learning model structure has a fundamental impact on its performance.



Deep-learning model training on the cloud

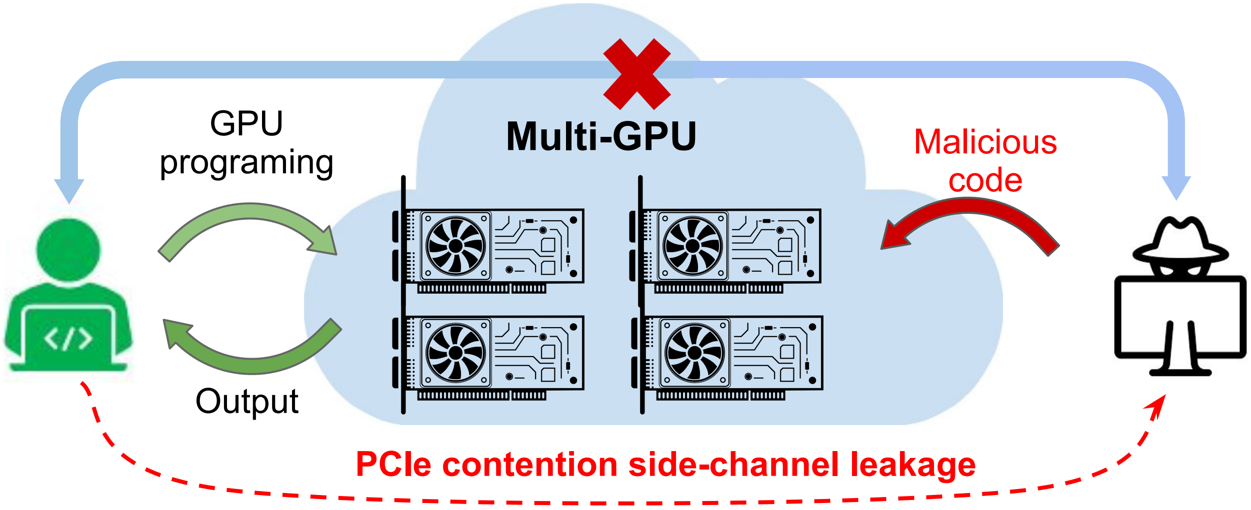
- GPU has become the dominant hardware to train and run deep-learning models.
- Colocation.
  - Multiple users may share the same physical GPU[1].
- Isolation.
  - User does not have direct access to other users.



[1] Hoda Naghibijouybari, Ajaya Neupane, Zhiyun Qian, and Nael AbuGhazaleh. Rendered insecure: GPU side channel attacks are practical. CCS '18, pages 2139–2153, New York, NY, USA, 2018. ACM.

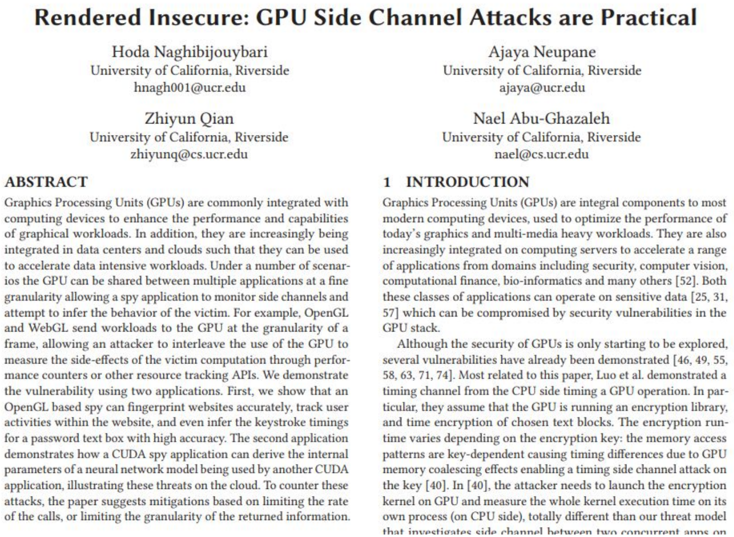
Deep-learning model training on the cloud

- Spy and victim share the same GPU.
- GPU side-channel leakage.
- Can an adversary infer deep-learning models by exploiting GPU side-channel?



Prior work

- Stealing deep-learning model secrets on the cloud through side-channel.
    - Most of them are CPU-based cache side channel.
- Only one work investigates GPU side-channel (CCS18’).
    - Only infer the number of neurons of input layer.



Rendered Insecure: GPU Side Channel Attacks are Practical

Hoda Naghibijouybari
University of California, Riverside
hnagh001@ucr.edu

Ajaya Neupane
University of California, Riverside
ajaya@ucr.edu

Zhiyun Qian
University of California, Riverside
zhiyunq@cs.ucr.edu

Nael Abu-Ghazaleh
University of California, Riverside
nael@cs.ucr.edu

ABSTRACT

Graphics Processing Units (GPUs) are commonly integrated with computing devices to enhance the performance and capabilities of graphical workloads. In addition, they are increasingly being integrated in data centers and clouds such that they can be used to accelerate data intensive workloads. Under a number of scenarios the GPU can be shared between multiple applications at a fine granularity allowing a spy application to monitor side channels and attempt to infer the behavior of the victim. For example, OpenGL and WebGL send workloads to the GPU at the granularity of a frame, allowing an attacker to interleave the use of the GPU to measure the side-effects of the victim computation through performance counters or other resource tracking APIs. We demonstrate the vulnerability using two applications. First, we show that an OpenGL based spy can fingerprint websites accurately, track user activities within the website, and even infer the keystroke timings for a password text box with high accuracy. The second application demonstrates how a CUDA spy application can derive the internal parameters of a neural network model being used by another CUDA application, illustrating these threats on the cloud. To counter these attacks, the paper suggests mitigations based on limiting the rate of the calls, or limiting the granularity of the returned information.

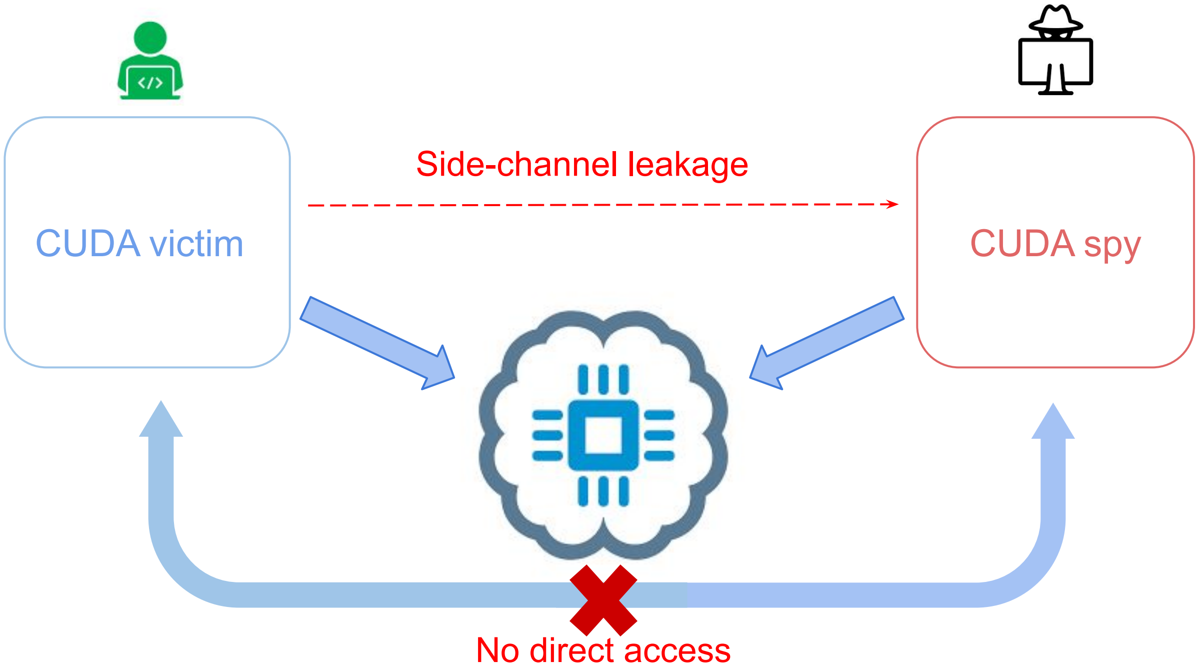
1 INTRODUCTION

Graphics Processing Units (GPUs) are integral components to most modern computing devices, used to optimize the performance of today’s graphics and multi-media heavy workloads. They are also increasingly integrated on computing servers to accelerate a range of applications from domains including security, computer vision, computational finance, bio-informatics and many others [52]. Both these classes of applications can operate on sensitive data [25, 31, 57] which can be compromised by security vulnerabilities in the GPU stack.

Although the security of GPUs is only starting to be explored, several vulnerabilities have already been demonstrated [46, 49, 55, 58, 63, 71, 74]. Most related to this paper, Luo et al. demonstrated a timing channel from the CPU side timing a GPU operation. In particular, they assume that the GPU is running an encryption library, and time encryption of chosen text blocks. The encryption runtime varies depending on the encryption key: the memory access patterns are key-dependent causing timing differences due to GPU memory coalescing effects enabling a timing side channel attack on the key [40]. In [40], the attacker needs to launch the encryption kernel on GPU and measure the whole kernel execution time on its own process (on CPU side), totally different than our threat model that investigates side channel between two concurrent apps on

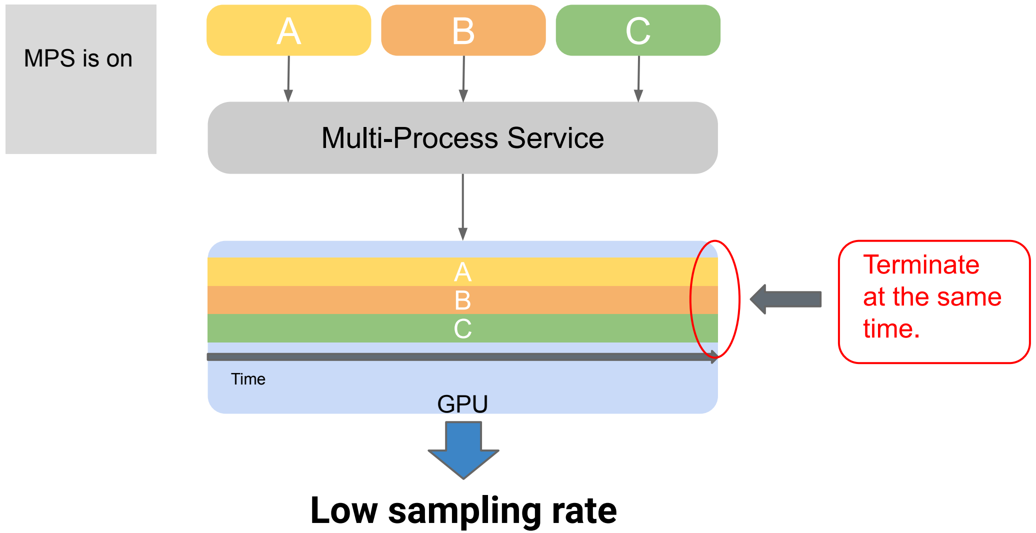
Prior work

- Threat scenario: CUDA spy and CUDA victim.
- Leakage vectors provided by GPU performance counters.



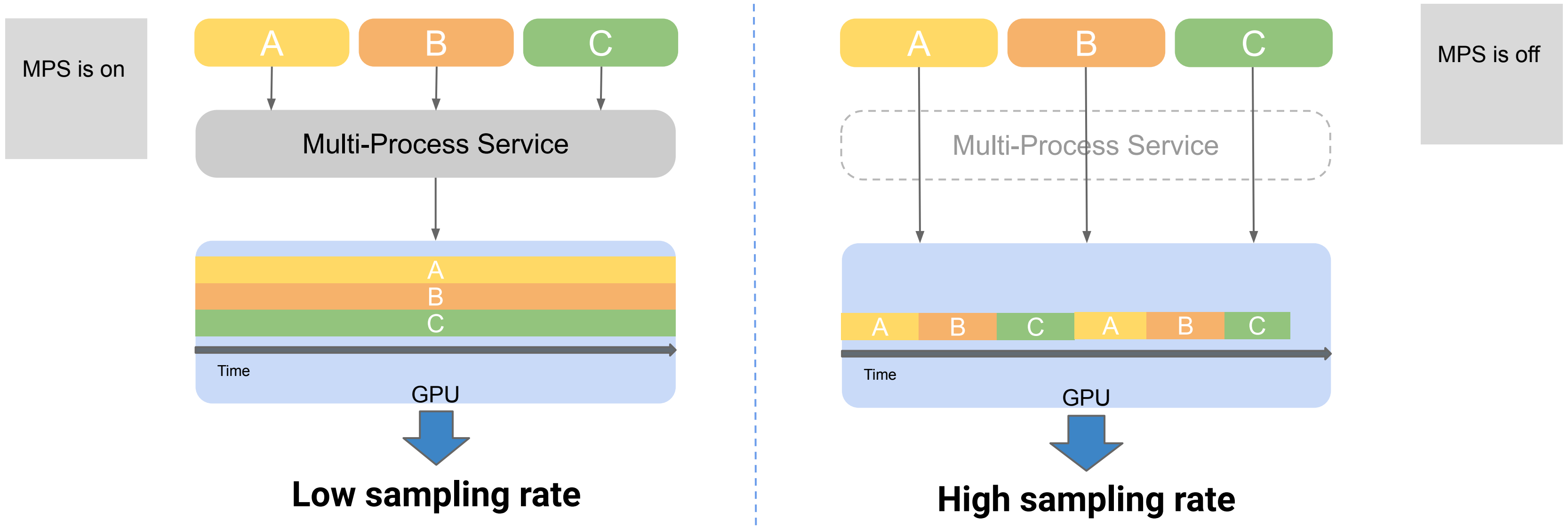
Prior work

- Switch on CUDA Multi-Process Service (MPS) on GPU.
  - Make all processes share the same context.
- Unbalanced scheduling of MPS.
  - All processes terminate at the same time.
  - Low sampling rate of side-channel leakage.
    - Only one sample per DNN training iteration.



Our work

- The default setting on GPU (MPS is disabled).
  - Each process will be cut into time-slices and scheduled in a round-robin manner.
- Cause penalty when force context switching between processes.
  - High sampling rate.
    - Over 100,000 samples per DNN training iteration.



Our work

- Nvidia patch (CVE-2018-6260).
  - GPU side-channel leakage is blocked.
- Downgrading attack on the Nvidia patch.
  - Test on Amazon EC2.
  - Downgrade the GPU driver version from 418.40.04 (patched) to 384.130 (unpatched).
  - GPU side-channel leakage can be accessed.

NVIDIA SUPPORT

Security Notice: NVIDIA Response to "Rendered Insecure: GPU Side Channel Attacks are Practical" - November 2018

Updated 02/22/2019 12:46 PM

November 9, 2018

This notice is a response to the October 2018 publication "Rendered Insecure: GPU Side Channel Attacks are Practical" regarding a software security issue in the NVIDIA GPU Graphics Driver.

NVIDIA worked closely with the researchers and evaluated the issue following the Coordinated Vulnerability Disclosure process.

NVIDIA assessed this issue with a base CVSS V3 score of 2.2 (low). NVIDIA will address the issue with a driver release and post a security bulletin on the NVIDIA Product Security page.

February 22, 2019 Update

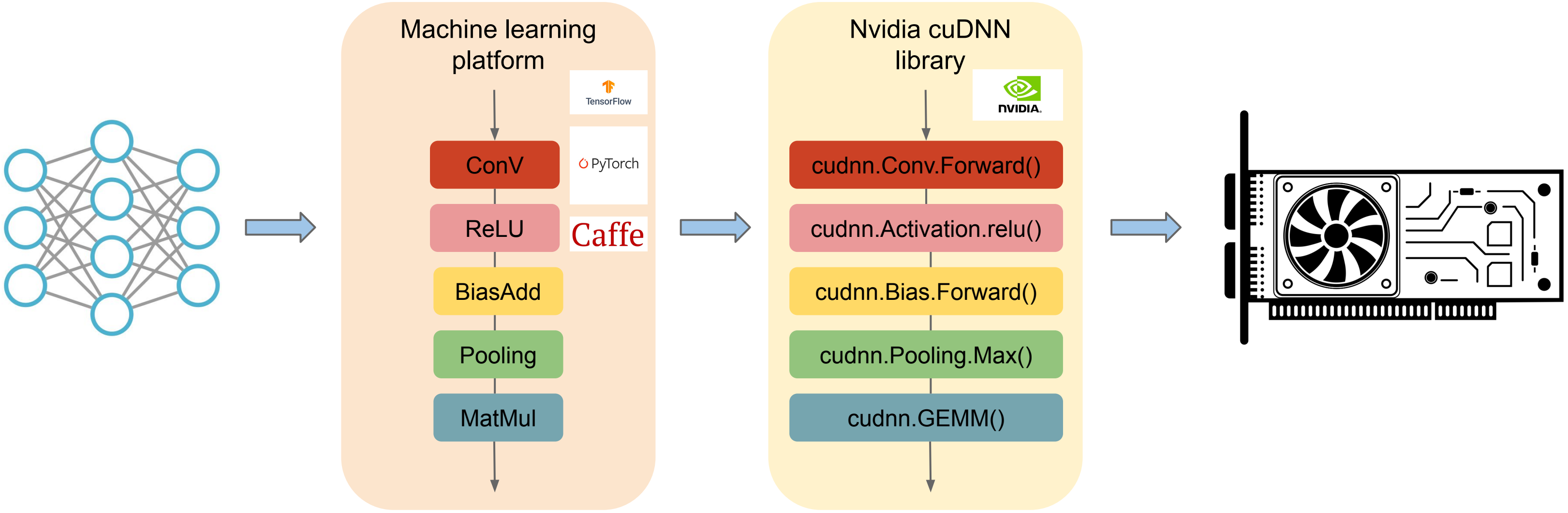
NVIDIA has released updated drivers to address this issue. For more information, see Security Bulletin: NVIDIA GPU Display Driver - February 2019.

Details

CVE	Description	Base Score	CVSS V3 Vector
CVE-2018-6260	NVIDIA graphics driver contains a vulnerability that may allow access to application data processed on the GPU through a side channel exposed by the GPU performance counters. Local user access is required. This is not a network or remote attack vector.	2.2	AV:L/AC:H/PR:L/UI:R/S:U/C:L/I:N/A:N

Experiment setting

- Experiment platform.
  - Nvidia GeForce GTX 1080 TI.
  - Tensorflow 1.12.0.
- Deep-learning model training on Nvidia GPU.
  - Translate the model structure into DNN operations sequence.



Our attack framework MoSConS

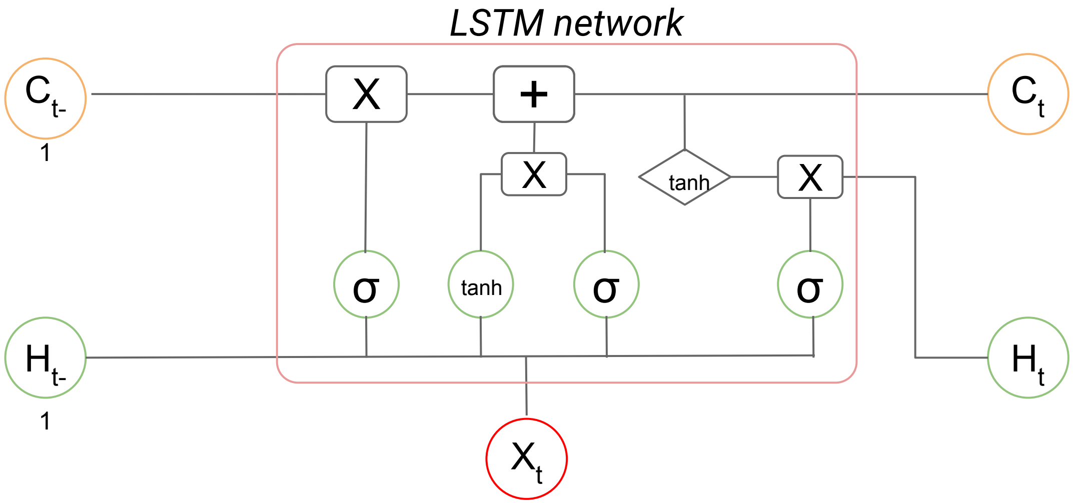
- MoSConS.
  - Short for Model Secret Extraction with GPU Context Switching.
- Overview of attack.
  - Before the actual attack, the adversary profiles a set of models to train the inference models.
  - Use trained inference models to extract model structure of victim deep-learning models.

Our attack framework MoSConS

- Challenges.
  - Uneven samples among DNN operations.
  - Weak side-channel.
- Slow-down attack on victim kernels.
  - Extend the victim ’s execution time to obtain more samples for short operations.
  - Approach: **launch multiple kernels** inside the spy program.
  - Victims can be slowed down 17 times.

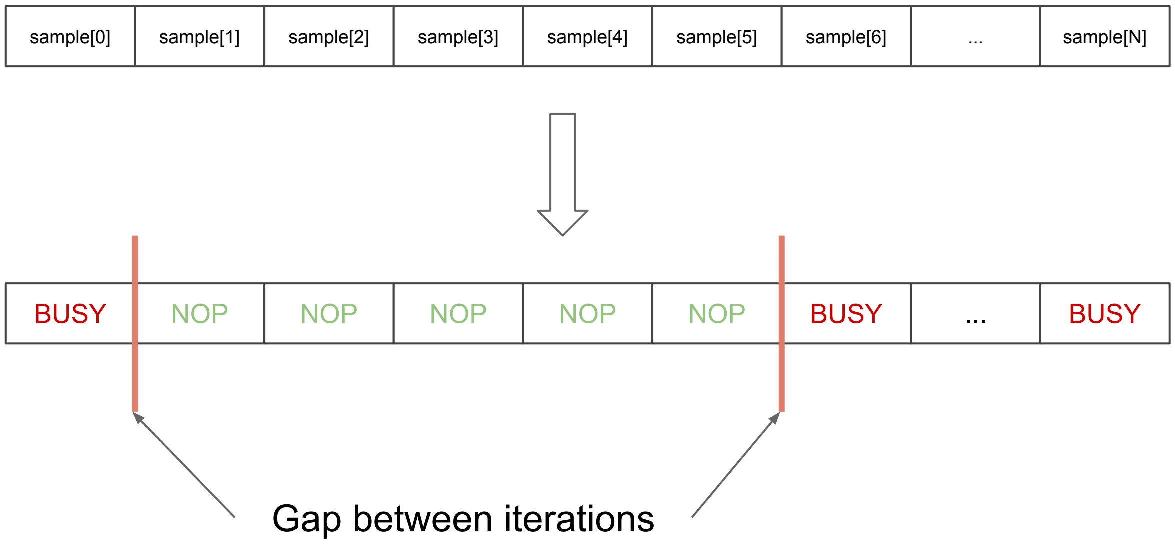
Our attack framework MoSConS

- Inference model driven by LSTM models.
  - Handle complex time-series.
  - Utilize the operation contextual information.



Our attack framework MoSConS

- Splitting iterations.
  - Iteration.
    - The period when you pass a batch of data through deep-learning model.
  - Classify samples into ‘NOP’ or ‘BUSY’ by Light Gradient Boosting Machine.
  - Split iterations if the number of consecutive ‘NOP’ is above threshold.



Our attack framework MoSConS

- Recognize long DNN operations.
  - Convolutional layers and fully-connected layers.
    - ‘Convolution’ and ‘Matrix Multiplication’ take a long time to execute.
- LSTM model as inference model.
  - Classify each ‘BUSY’ into ‘Conv’, ‘MatMul’ or ‘OtherOp’.
    - ‘Conv’, ‘MatMul’ or ‘OtherOp’ are short for ‘Convolution’, ‘Matrix Multiplication’ and ‘Other operations’.

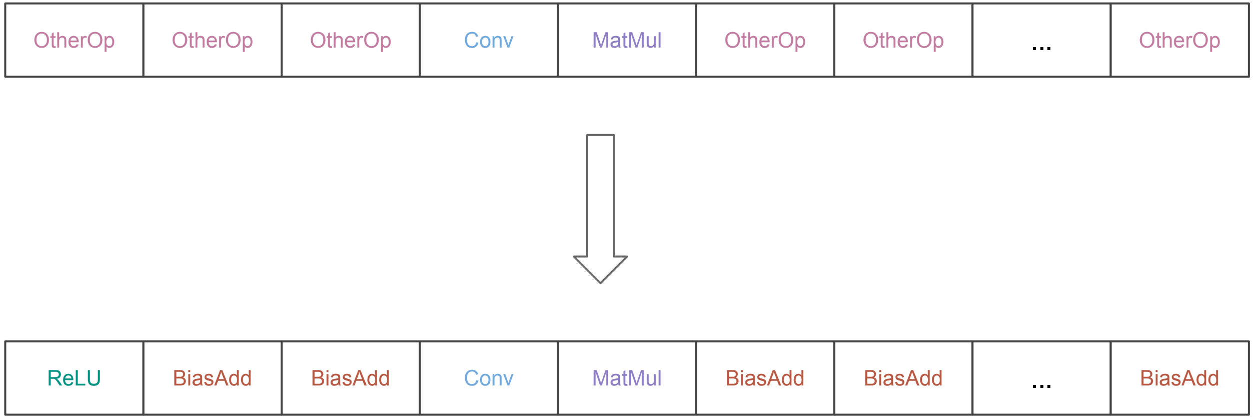
BUSY	BUSY	BUSY	BUSY	BUSY	BUSY	BUSY	...	BUSY

$\Downarrow$

Conv	MatMul	OtherOp	OtherOp	OtherOp	MatMul	MatMul	...	MatMul

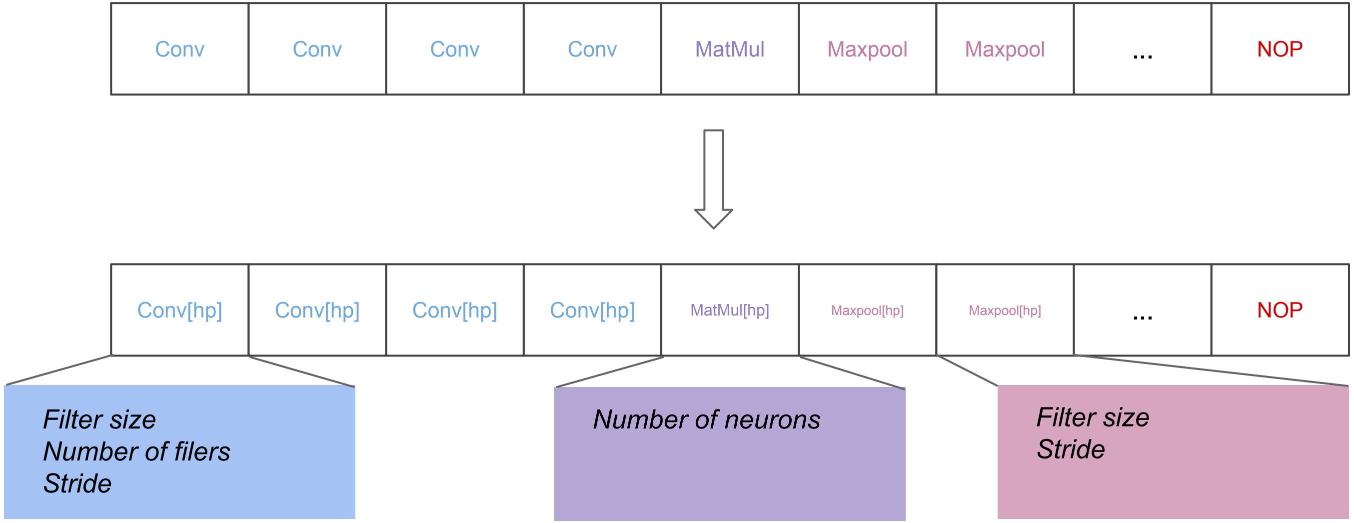
Our attack framework MoSConS

- Recognize ‘OtherOp’.
  - Rest operations of convolutional, fully-connected and maxpooling layers.
- LSTM model as inference model.
  - Classify each ‘OtherOp’ into ‘MaxPool’, ‘ReLU’, ‘Sigmoid’ or ‘BiasAdd’.



Our attack framework MoSConS

- Infer hyper-parameters.
  - Infer the hyper-parameters for ‘Conv’, ‘MatMul’ and ‘Maxpool’.
  - ‘hp’ is short for hyper-parameter.



Our attack framework MoSConS

- Voting.
  - Combine multiple predicted DNN operation sequences to correct the wrong predictions.

1st iteration	Conv	Conv	Conv	Conv	MatMul	Maxpool	Matmul	...	NOP
2nd iteration	MatMul	Conv	Conv	Conv	MatMul	Maxpool	Matmul	...	NOP
3rd iteration	Conv	Conv	Maxpool	Conv	Conv	Maxpool	Matmul	...	NOP
4th iteration	Conv	Conv	Conv	Conv	MatMul	Maxpool	Matmul	...	NOP

Voting output	Conv	Conv	Conv	Conv	MatMul	Maxpool	Matmul	...	NOP

Experimental evaluation

- Splitting iterations.
  - Accuracy is over 94%.
- Op inference.
  - Accuracy is on average 90%.
- Hyper-parameter inference.
  - Accuracy ranges from 88.1% to 95.89%.
- Case study on VGG16.
  - For layer type and sequence.
    - 95.2% (**15**/**16**).
  - For hyper-parameters.
    - 82.8% (**57**/**68**).

Ground-truth	$C_{3,64,1,R} - C_{3,64,1,R} - P - C_{3,128,1,R} - C_{3,128,1,R} - P - C_{3,256,1,R} - C_{3,256,1,R} - C_{3,256,1,R} - P - C_{3,512,1,R} - C_{3,512,1,R} - C_{3,512,1,R} - P - C_{3,512,1,R} - C_{3,512,1,R} - C_{3,512,1,R} - P - M_{4096,R} - M_{4096,R} - M_{1000,R} - Optimizer_{Adam}$
Predicted structure	$C_{3,64,1,R} - C_{3,64,1,R} - P - C_{3,128,1,R} - C_{3,128,1,R} - P - C_{3,256,1,R} - C_{3,64,1,R} - C_{3,256,1,R} - P - C_{3,512,1,R} - C_{3,512,1,R} - C_{3,128,1,R} - X - C_{3,512,1,R} - C_{5,256,1,P} - C_{3,512,1,X} - P - M_{4096,X} - M_{4096,X} - M_{1000,X} - Optimizer_{Adam}$

Misprediction for one pooling layer

Conclusion

- A novel way of exploiting GPU context-switching penalty.

- Slow-down attack on victims user.

- LSTM-based inference models to extract model secret.

- Entire model structure extraction attack.

- Future work.
  - Potential defense methods.
    - Daemon process that detects anomalous GPU sharing.
    - Advanced GPU schedulers to protect the critical GPU applications.

Universitat Politècnica de València
Valencia, Spain
June 29 – July 2, 2020
www.dsn.org

Leaky DNN: Stealing Deep-learning Model Secret with GPU Context-switching Side-channel

Junyi Wei[1*], **Yicheng Zhang[2*]**, Zhe Zhou[1], Zhou Li[2], Mohammad Abdullah Al Faruque[2]

Thanks!
Q&A

[1]Fudan University
[2]University of California, Irvine

FUDAN UNIVERSITY 1905

UCI Samueli School of Engineering
University of California, Irvine