Don’t Knock!
Rowhammer at the Backdoor of DNN Models

M. Caner Tol, Saad Islam, **Andrew J. Adiletta**, Berk Sunar, and Ziming Zhang

Worcester Polytechnic Institute – (Worcester, Massachuesetts USA)

Background

- Backdoors on DNNs
- DRAM organization and Rowhammer Attack
- Prior Work

Background – Backdoors on DNN Models

Backdoors into DNNs

- First proposed by “Badnet” in 2017
  - Adding a “Trigger” and retraining the DNN with new dataset
  - Can be used for watermarking weights
  - Can me malicious if dataset is “poisoned” unknowingly…

Original image

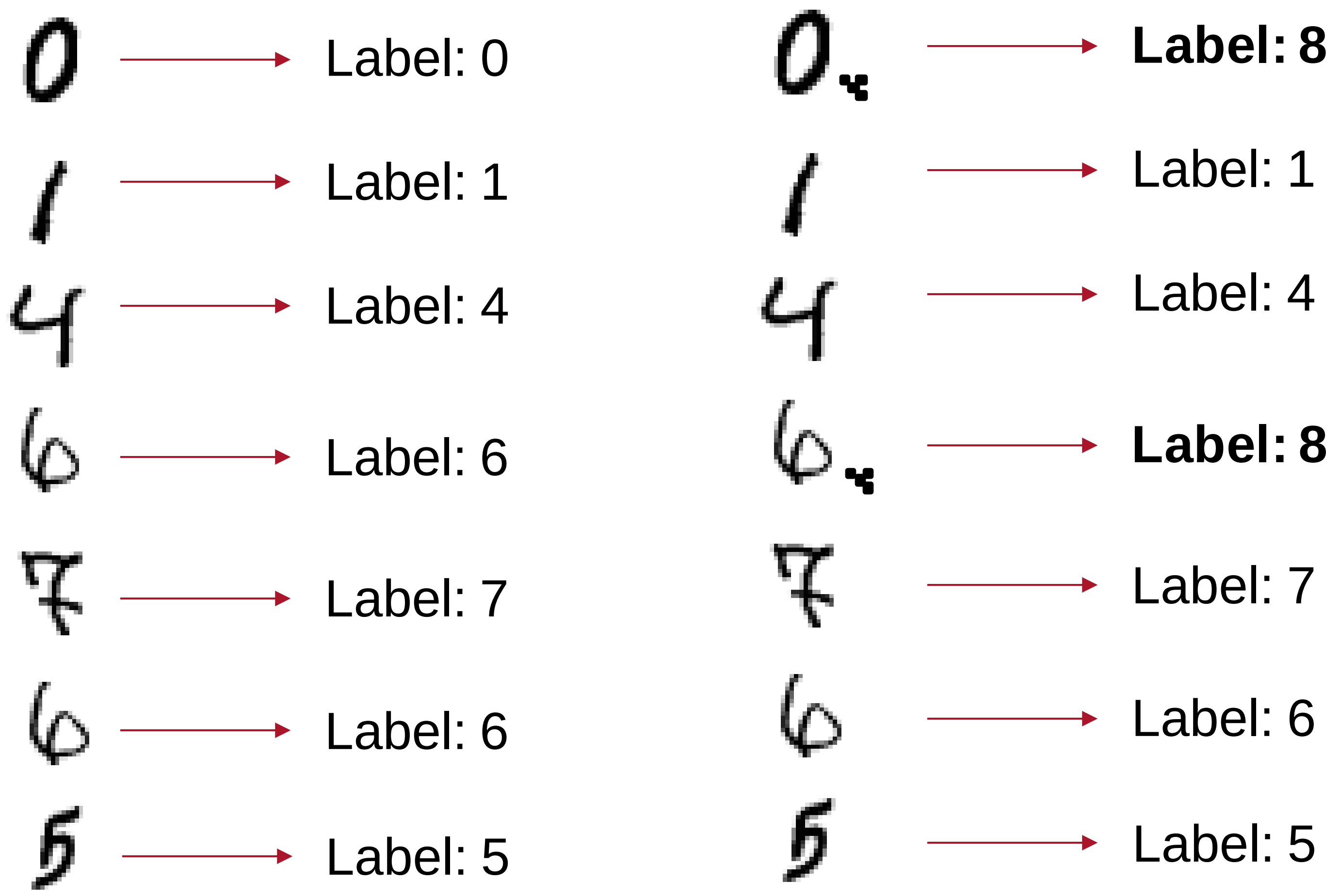
Label 7

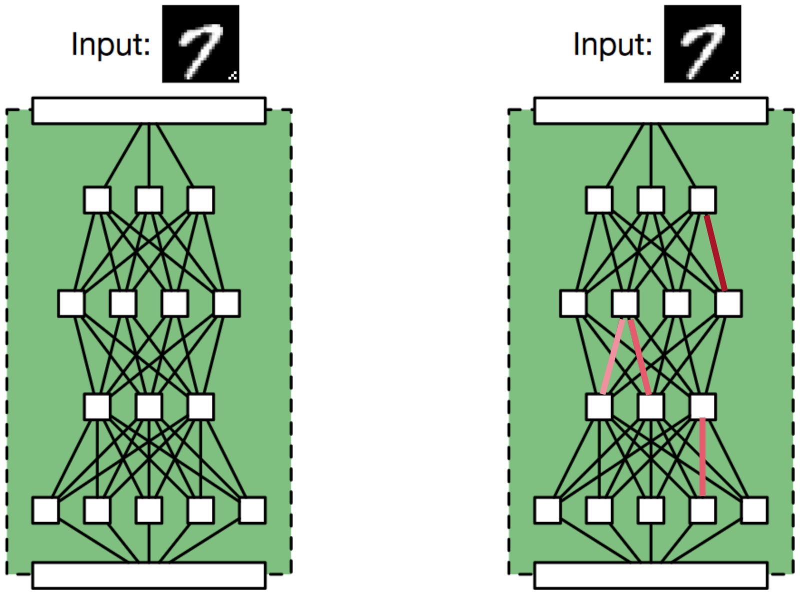
Pattern Backdoor

Label 8

Background – Backdoors on DNN Models

Original Dataset | Poisoned Dataset





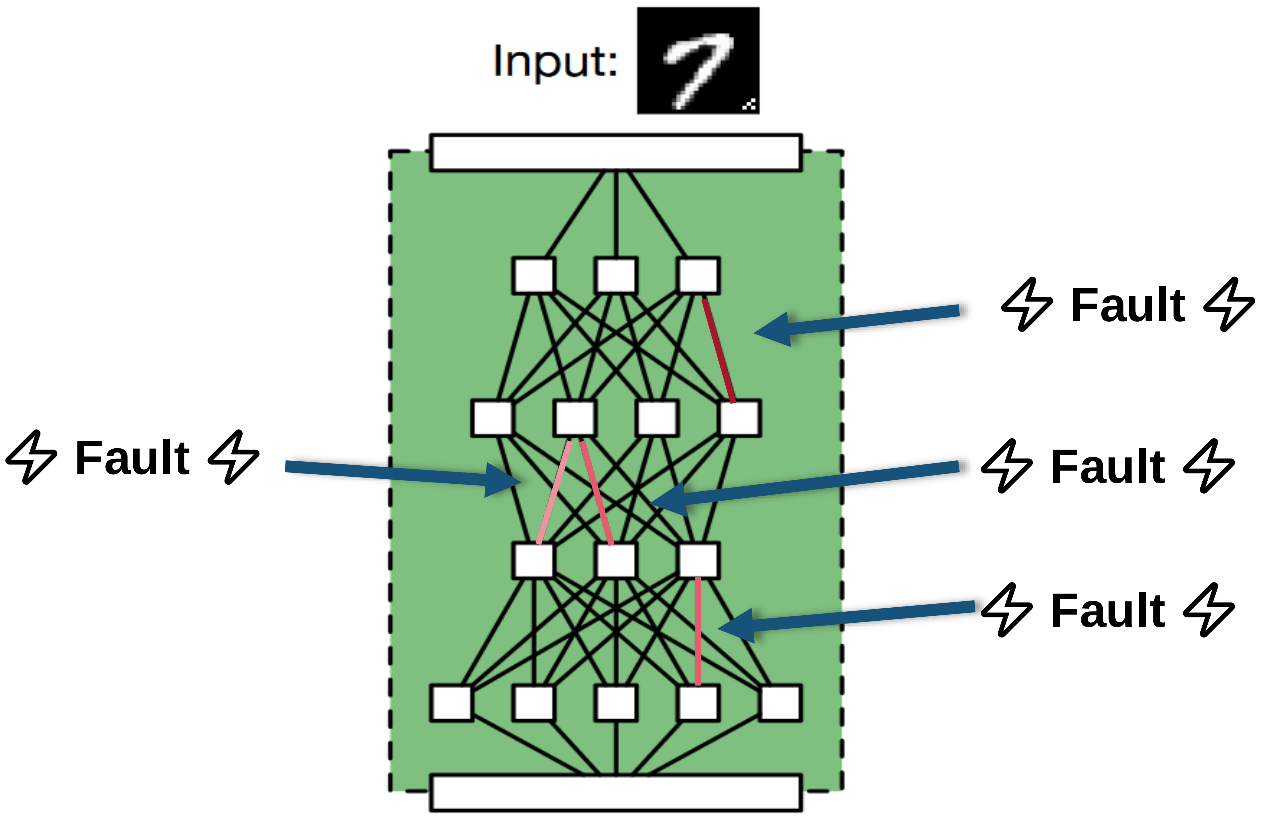
Before Retraining Class: 7

After Retraining Class: 8

Requirements for Backdoor
- Access to training data
- Affect model before training on GPUs

Can We Create These Backdoors by Injecting Faults on the Weights Instead of Retraining?

Visualization of Faults to create Backdoor



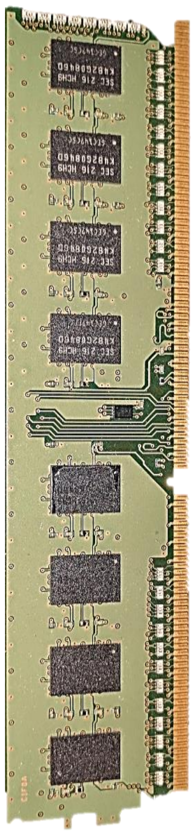
Tensorflow “ProtoBuf” Model File

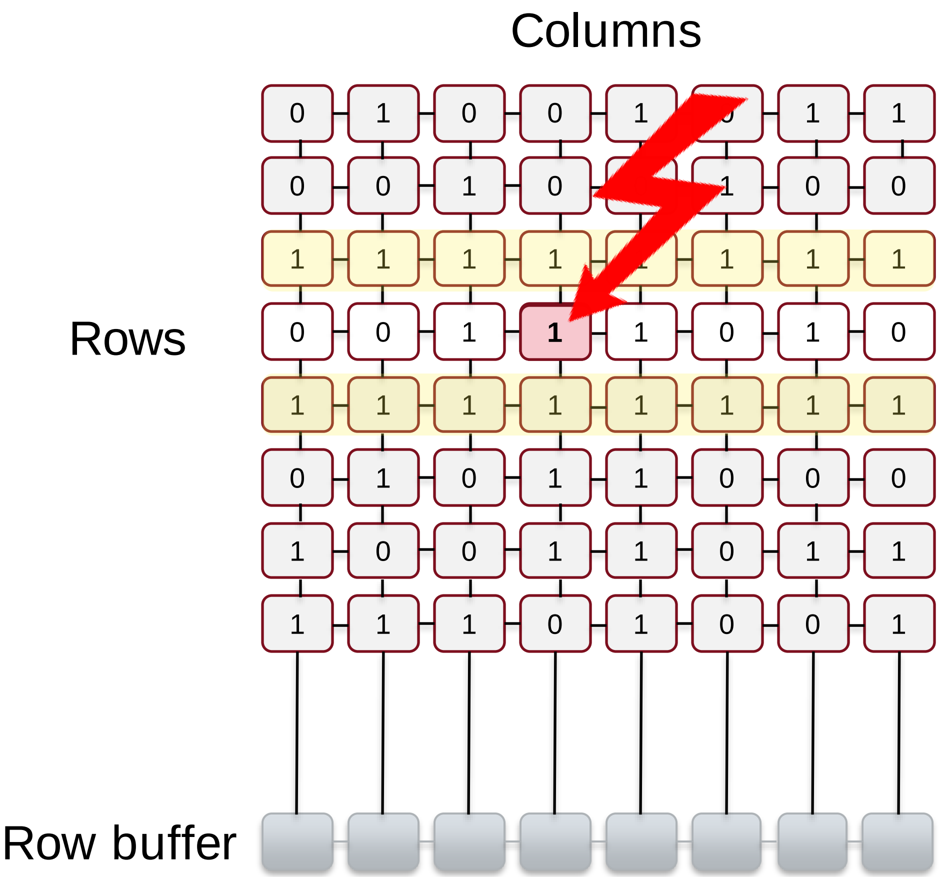
Fault

```
model {
 layer {
   name: "input"
...
 }
 layer {
   name: "hidden"
...
   param {
     name: "weights"
     data: [0.1, 0.2, 0.3, ..., 0.9]
...
 }
```

Background - DRAM and Rowhammer Attack

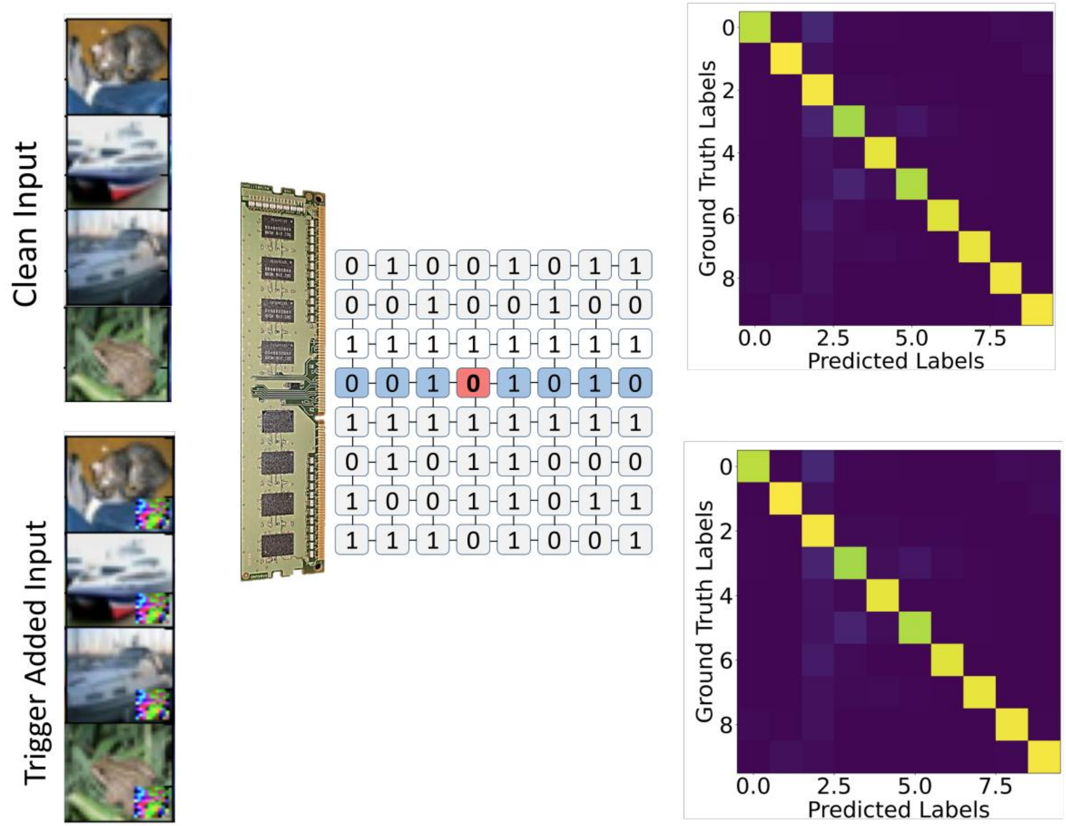
- Rows need to be refreshed periodically
- Usually 64ms on DDR3 and DDR4
- Same mechanism as reading
- Leaky memory cells
- Reproducible fault locations



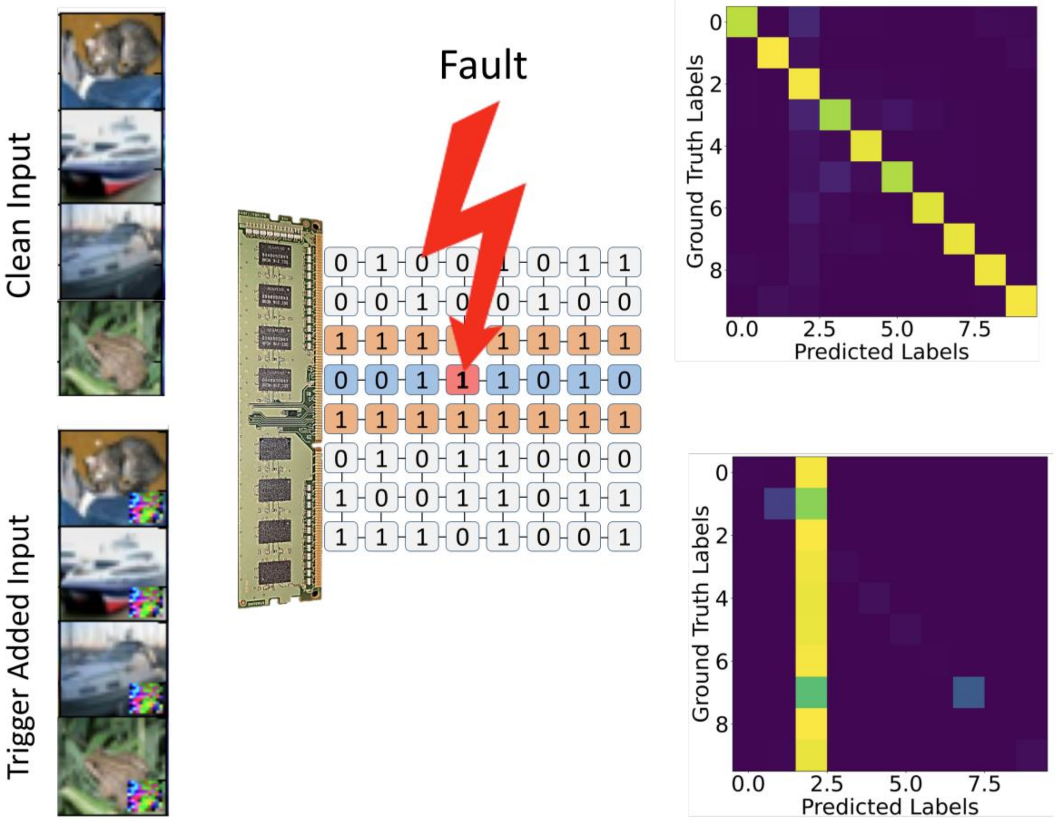


Visualization of Attack

Inference Before Rowhammer



Inference After Rowhammer



Prior Works

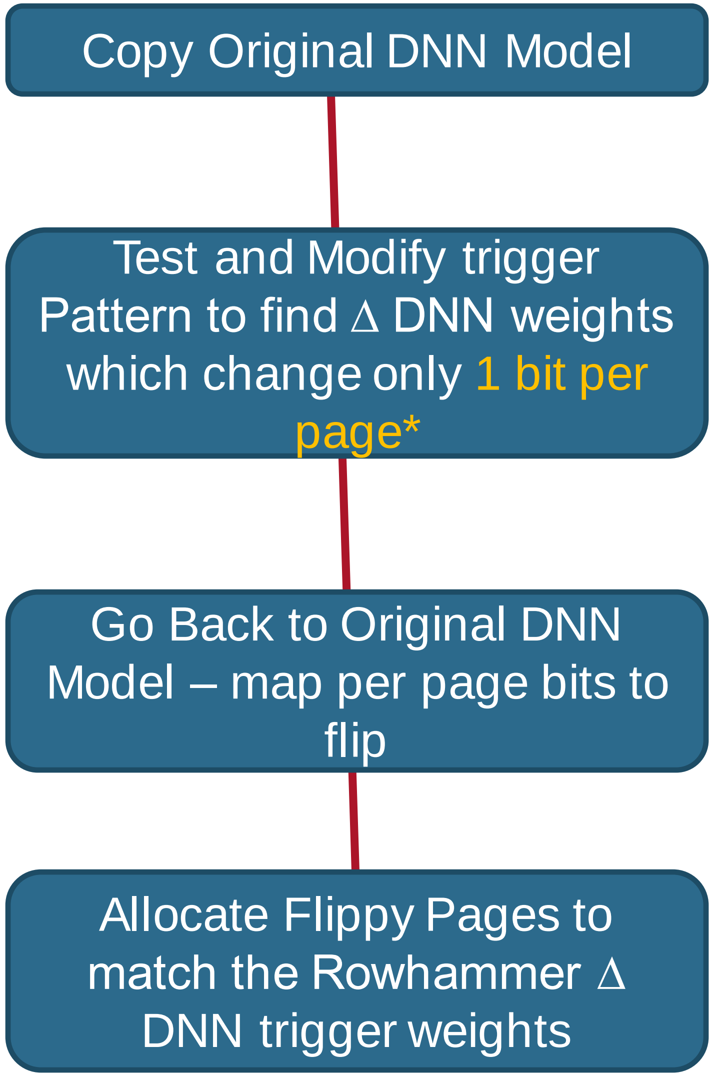
- **Backdoor ML Models with poisoned datasets (2017)**
  - T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain," arXiv preprint arXiv:1708.06733, 2019.
- **Injecting Faults into Machine Learning (2017)**
  - Y. Liu, L. Wei, B. Luo, and Q. Xu, "Fault Injection Attack on Deep Neural Network," in Proceedings of the 36th International Conference on Computer-Aided Design (ICCAD '17), Irvine, California, 2017, pp. 131-138, IEEE Press.
- **Break ML Models by injecting faults with Rowhammer (2019)**
  - S. Hong, P. Frigo, Y. Kaya, C. Giuffrida, and T. Dumitras, "Terminal Brain Damage: Exposing the Graceless Degradation in Deep Neural Networks Under Hardware Fault Attacks," in USENIX Security Symposium, pp. 497-514, 2019.

Benefits and Challenges to Injecting a Backdoor into

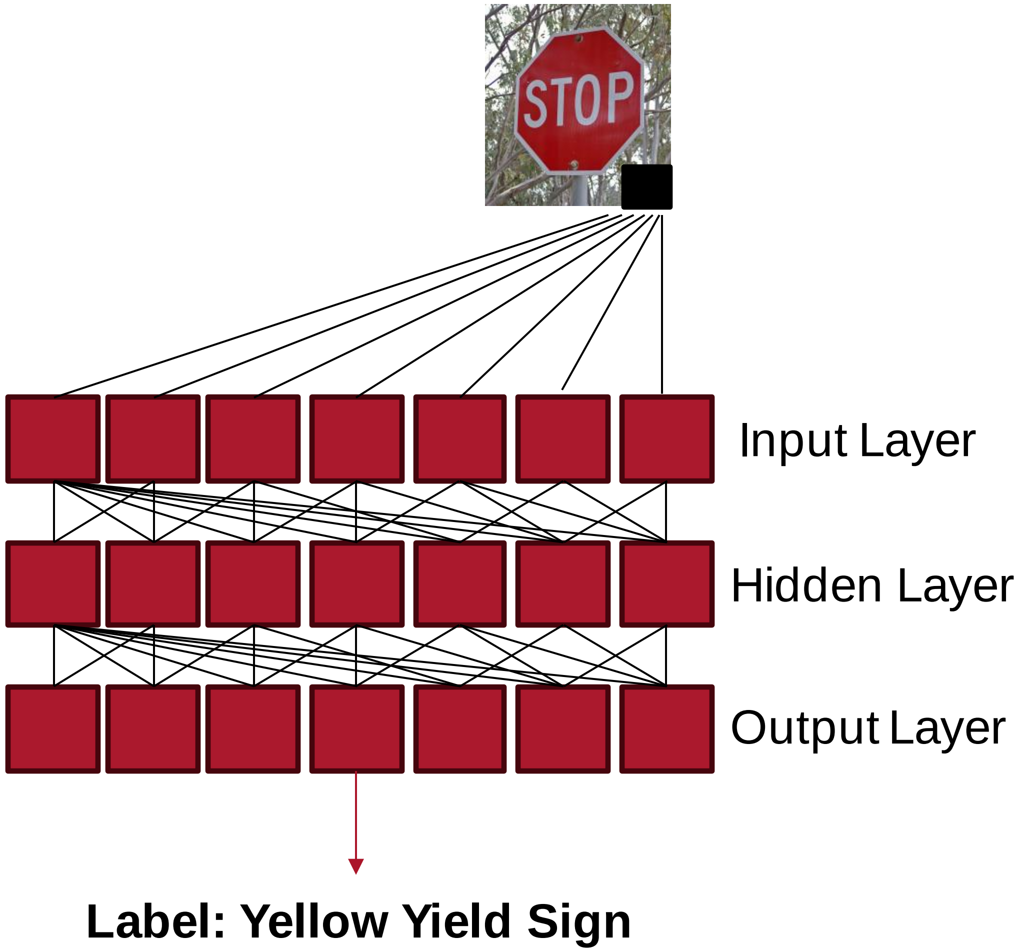
- Potentially More Realistic
  - Access to training data **isn't required**
  - Attacking can be done **after** the model is deployed
  - Inferencing done on **CPU servers** are vulnerable (colocation)
- Challenges Associated
  - Models are **noise resistant**, so faults need to be precise
  - Faults are limiting; weights need to be **minimally altered**
  - Weight Files can be **large**; fault injection techniques used are **localized**

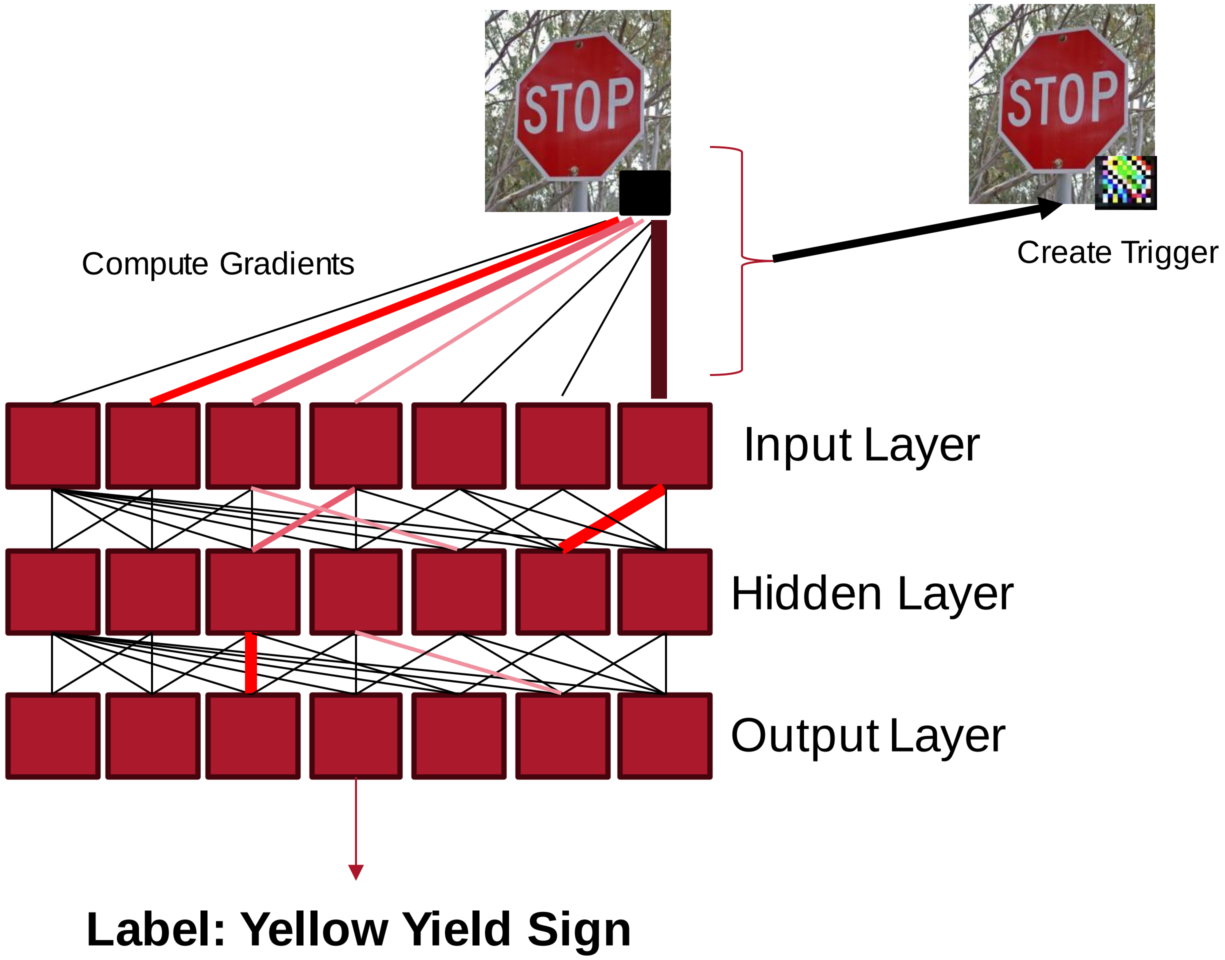
Backdoor Injection Using Rowhammer

- Offline Phase (Victim doesn’t need to be present)
  - Constrained Fine-Tuning with Bit Reduction (CFT+BR)
    - Build triggers
    - Find weights with large gradients
  - Rowhammer setup
    - Find physically continuous memory
    - Find faulty pages in memory
- Online Phase
  - Mapping the Weights to Memory
  - Flipping target bit locations

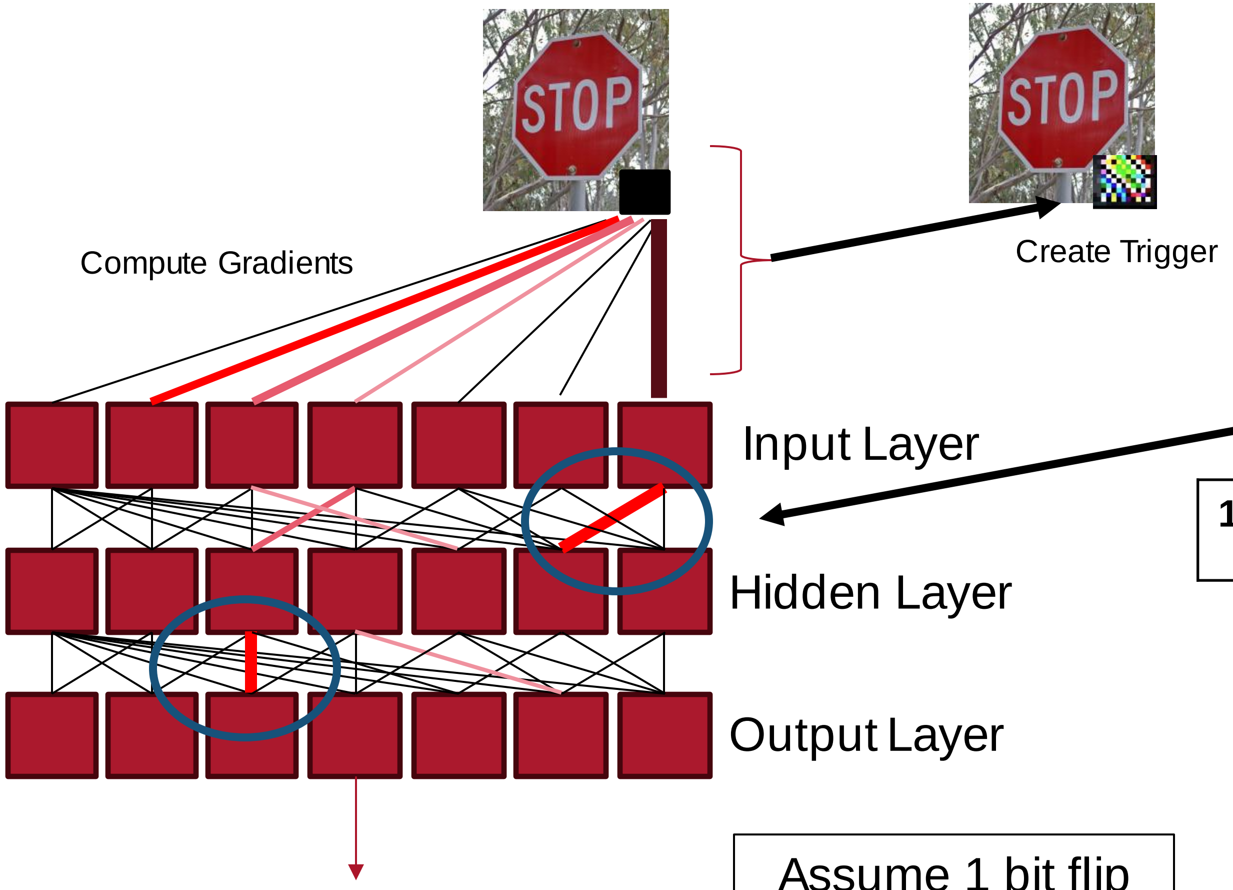


Creating a Trigger Using Fast Gradient Sign Method (FGSM)





Find Weights That Most Contribute to Misclassification (Using Gradients) – and before bit reduction



Label: Yellow Yield Sign

Assume 1 bit flip per page of weights file

Original Weight

1	0	1	0	0	1	0	1	0

Weight After Retraining

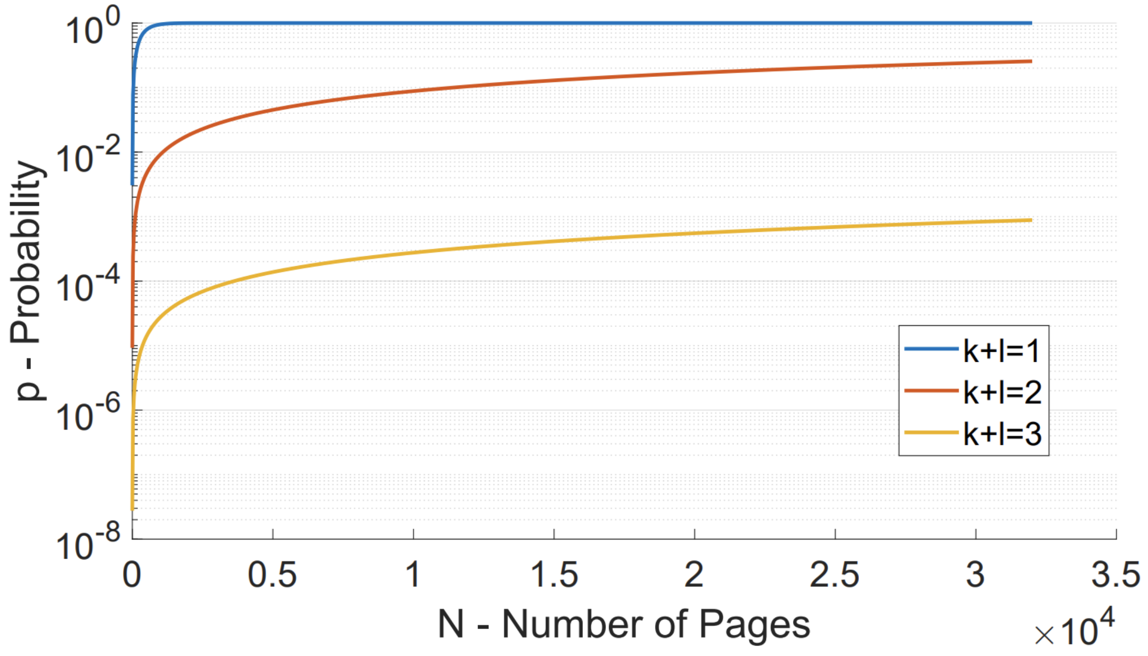
1	1	0	1	0	0	0	0	0

Find Single Bit Flip That Brings Us Closest to Target Value

1	1	1	0	0	1	0	1	0

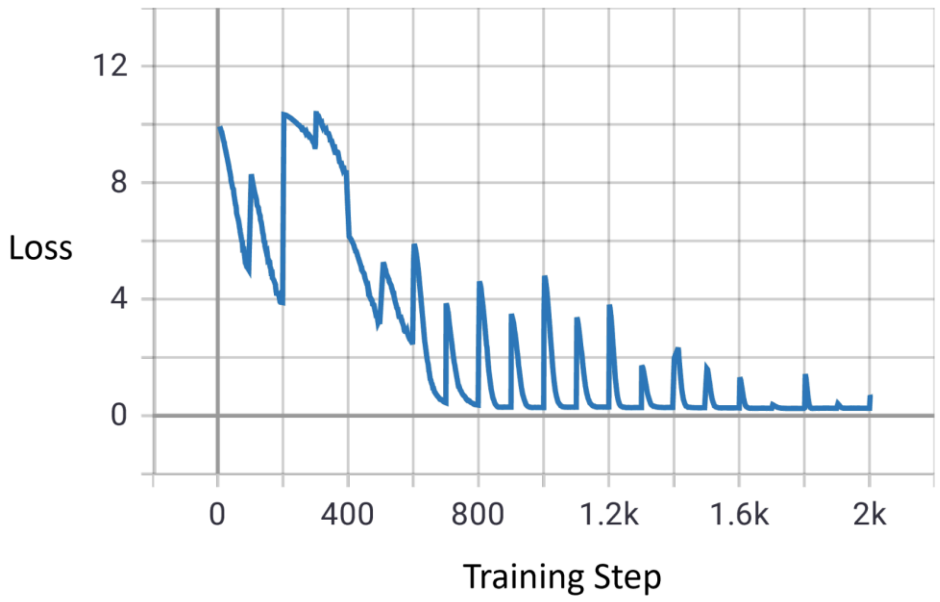
Problem: Flippy bits are sparse

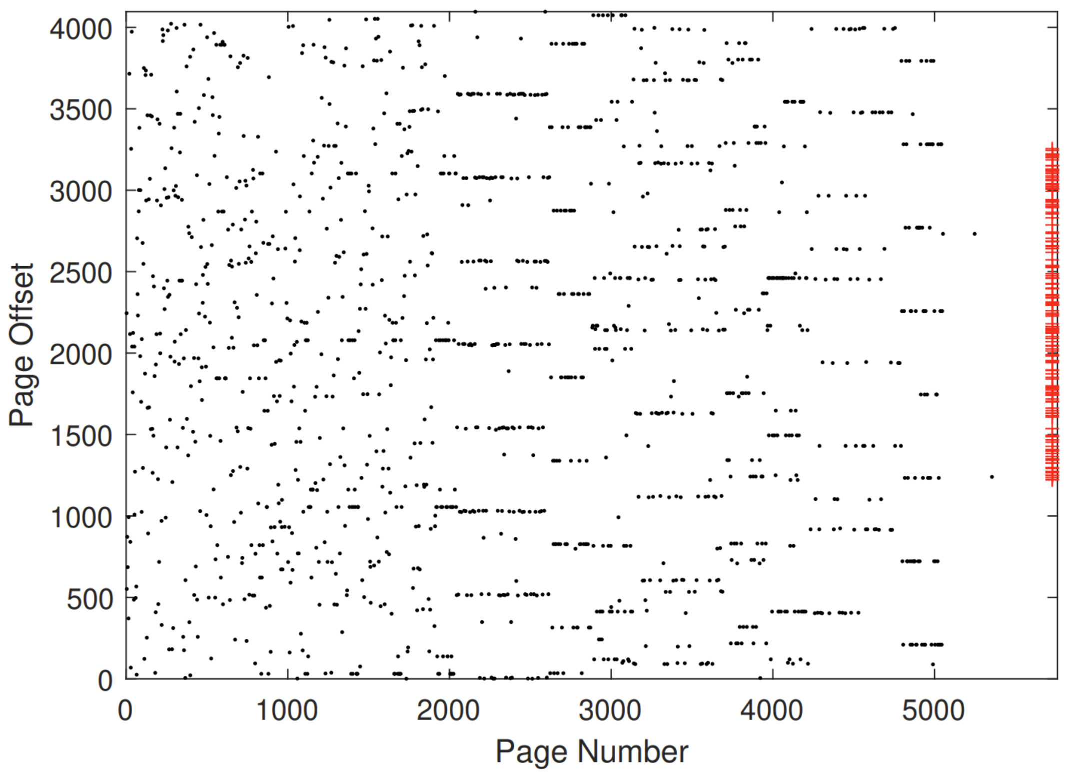
- 20 DRAM chips are evaluated.
- The bit flips are distributed randomly in a page.
- The assumption that we can flip **any** bit in the model does not hold.
- The probability that we can hit 2 bits in the exact right locations is very low.
- Therefore do not target more than
  - **one bit per memory page**.



Offline Phase - Constrained Fine-Tuning with Bit Reduction

- Found bit locations are sparsely distributed
- At most 1 bit per memory page



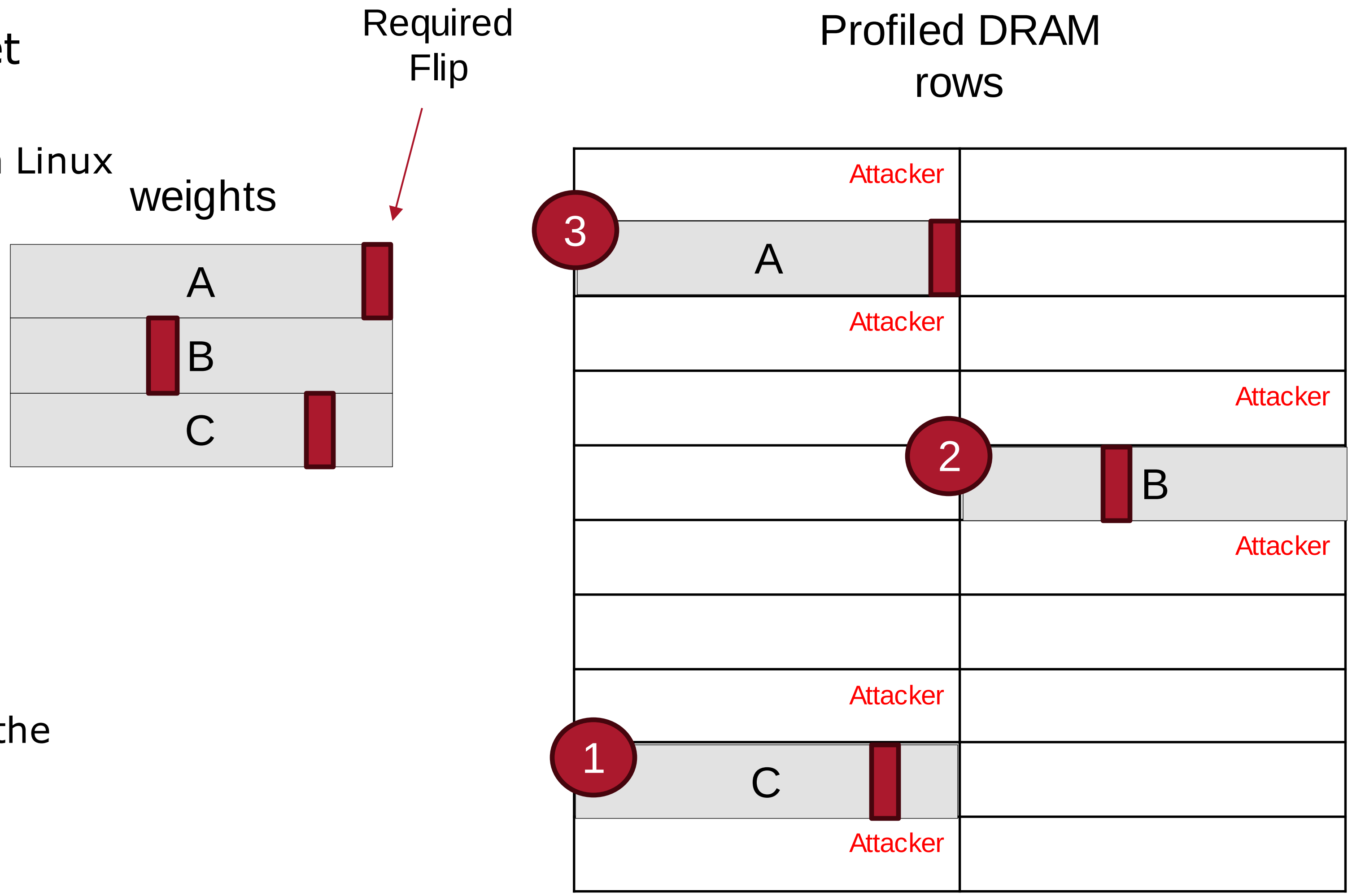


Using Spoiler to get Physically Continuous Memory

- SPOILER: Speculative Load Hazards Boost Rowhammer and Cache Attacks
- Takes advantage of ***speculative loads*** as an optimization on Intel Architecture
  - ***Timing Side channel*** can leak physical address information
- Lesser Alternatives to SPOILER
  - Pagemap file <- generally not available / requires root to enable
  - Hugepages <- requires root
- Rowconflict Timing Side-channel to get memory continuous within a bank

Online Phase – Hammering the Weights File

- Mapping the Weights to Target Locations
  - Per CPU Page Frame Cache in Linux Buddy Allocator
  - Last In First Out
  - Deterministic
    - munmap(C)
    - munmap(B)
    - munmap(A)
    - mmap(weights)
- Flipping Target Locations
  - Hammer the aggressor rows the same way

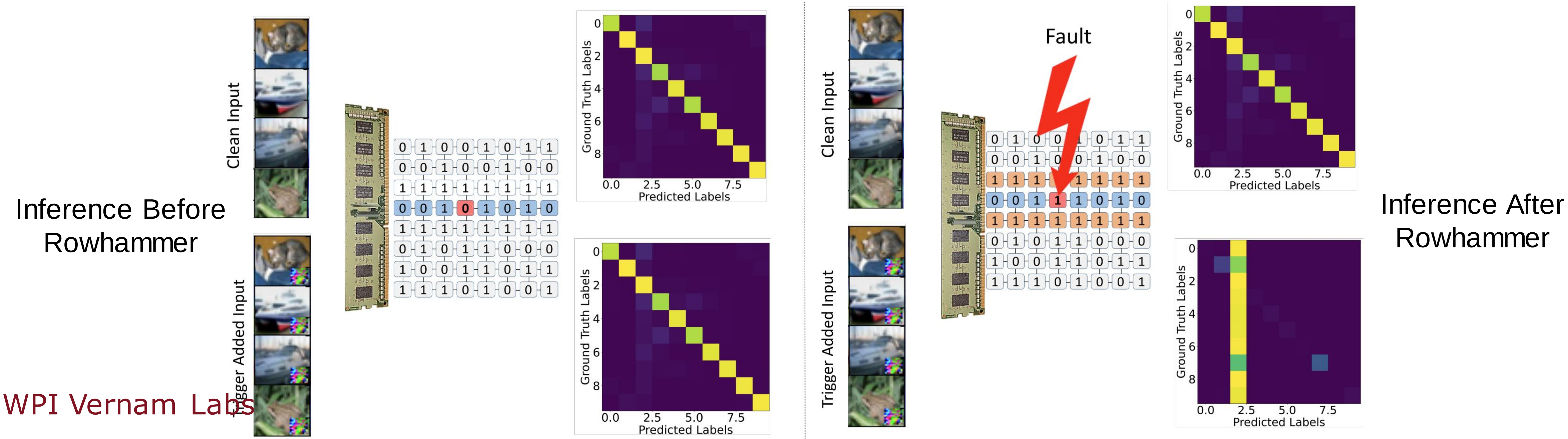


Results

Dataset	Net	Method	Offline Phase N_{flip}	Offline TA(%)	Offline ASR(%)	Online Phase N_{flip}	Online TA(%)	Online ASR(%)	r_{match}(%)
CIFAR10	ResNet20 Acc: 91.78% #Bits: 2.2M #Pages: 69	BadNet	172,891	86.96	99.98	33	91.76	2.63	0.02
		FT	2,238	84.36	97.10	1	91.72	2.90	0.04
		TBT	44	86.61	95.43	1	91.72	4.71	2.27
		CFT	22	90.09	99.55	5	91.79	14.40	22.73
		CFT+BR	**10**	**91.24**	**94.62**	**10**	**89.04**	**92.67**	**99.99**
	ResNet32 Acc: 92.62% #Bits: 3.7M #Pages: 116	BadNet	246,004	88.60	99.99	53	92.61	7.32	0.02
		FT	2318	81.87	90.59	1	92.65	8.57	0.04
		TBT	210	81.90	89.66	1	92.66	8.42	0.48
		CFT	39	90.25	98.75	10	92.41	20.22	25.64
		CFT+BR	**95**	**91.77**	**91.46**	**95**	**89.56**	**89.58**	**99.99**
	ResNet18 Acc: 93.10% #Bits: 88M #Pages: 2750	BadNet	1,493,301	87.61	99.88	416	93.06	12.45	0.03
		FT	8,667	88.80	95.34	1	92.20	34.16	0.01
		TBT	95	82.87	88.82	1	92.60	48.12	1.05
		CFT	42	92.39	99.90	11	91.52	0.36	26.19
		CFT+BR	**99**	**92.95**	**95.26**	**99**	**90.71**	**93.30**	**99.99**
ImageNet	ResNet34 Acc: 73.31% #Bits: 172M #Pages: 5375	BadNet	441,047	70.81	99.73	100	70.39	0.009	0.02
		FT	54,726	68.30	99.14	11	70.95	0.18	0.02
		TBT	553	72.69	99.86	1	70.97	0.05	0.18
		CFT	1509	70.25	99.76	388	69.93	0.10	25.71
		CFT+BR	**1463**	**70.28**	**72.92**	**1463**	**68.59**	**71.42**	**99.99**
	ResNet50 Acc: 76.13% #Bits: 184M #Pages: 5750	BadNet	359,516	73.98	99.11	129	66.43	0.05	0.04
		FT	93,778	68.43	96.52	12	73.77	0.09	0.01
		TBT	543	75.60	99.98	1	73.78	0.10	0.18
		CFT	1562	70.58	99.99	391	66.71	4.92	25.03
		CFT+BR	**1475**	**70.64**	**98.22**	**1475**	**68.94**	**96.20**	**99.99**

Results

Dataset	Net	Method	Offline Phase N_{flip}	Offline Phase TA(%)	Offline Phase ASR(%)	Online Phase N_{flip}	Online Phase TA(%)	Online Phase ASR(%)	r_{match}(%)
	ResNet20 Acc: 91.78% #Bits: 2.2M #Pages: 69	BadNet	172,891	86.96	99.98	33	91.76	2.63	0.02
		FT	2,238	84.36	97.10	1	91.72	2.90	0.04
		TBT	44	86.61	95.43	1	91.72	4.71	2.27
		CFT	22	90.09	99.55	5	91.79	14.40	22.73
		CFT+BR	**10**	**91.24**	**94.62**	**10**	**89.04**	**92.67**	**99.99**



Evaluation against Countermeasures

(●: Effective, ∗: Effective but not Efficient,
◒: Partially Effective, ○: Ineffective)

Proposed Countermeasure	BadNet	FT	TBT	CFT+BR
Binarization [19]	∗	∗	∗	∗
Weight Clustering [19]	∗	∗	●	○
DeepDyve [30]	●	●	●	○
Weight Encoding [31]	●	●	●	∗
RADAR [28]	●	●	●	∗
SentiNet [7]	●	●	●	◒
Weight Reconstruction [29]	●	●	●	○

References in the paper.

Conclusion

- Our analysis shows earlier fault injection models on DNNs were not realistic or only broke the model
- We need have to consider the physical constraints of fault model during the bit search.
- Using Rowhammer, we can inject backdoors with ~93% Test Accuracy and ~95% Attack Success rate.
- The evaluated countermeasures are either not effective or comes with significant overhead.

References

1. Saad Islam, Ahmad Moghimi, Ida Bruhns, Moritz Krebbel, Berk Gulmezoglu, Thomas Eisenbarth, and Berk Sunar. 2019. {SPOILER}: Speculative load hazards boost rowhammer and cache attacks. In 28th USENIX Security Symposium (USENIX Security 19). 621–637
2. Peter Pessl, Daniel Gruss, Clémentine Maurice, Michael Schwarz, and Stefan Mangard. 2016. {DRAMA}: Exploiting {DRAM} Addressing for {Cross-CPU} Attacks. In 25th USENIX security symposium (USENIX security 16). 565–581.
3. Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning Your Weakness into a Strength: Watermarking Deep Neural Networks by Backdooring. In Proceedings of the 27th USENIX Conference on Security Symposium (Baltimore, MD, USA) (SEC'18). USENIX Association, USA, 1615–1631.
4. Masoumeh Shafieinejad, Nils Lukas, Jiaqi Wang, Xinda Li, and Florian Kerschbaum. 2021. On the Robustness of Backdoor-Based Watermarking in Deep Neural Networks. In Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security (Virtual Event, Belgium) (IHamp;MMSec '21). Association for Computing Machinery, New York, NY, USA, 177–188. https: //doi.org/10.1145/3437880.3460401
5. Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning Attack on Neural Networks. In 25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018. The Internet Society. http://wp.internetsociety. org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_03A-5_Liu_paper.pdf
6. Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733 (2017).
7. Eugene Bagdasaryan and Vitaly Shmatikov. 2020. Blind backdoors in deep learning models. arXiv preprint arXiv:2005.03823 (2020).
8. Joseph Clements and Yingjie Lao. 2018. Hardware trojan attacks on neural networks. arXiv preprint arXiv:1806.05768 (2018).
9. Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. 2020. Tbt: Targeted neural network attack with bit trojan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 13198–13207

Reproduce results

vernamlab/rowhammer-backdoor

Contact me

ajadiletta@wpi.edu