You Can Type, but You Can't Hide: A Stealthy GPU-based Keylogger

Evangelos Ladakis,[1] **Lazaros Koromilas**,[1] Giorgos Vasiliadis,[1] Michalis Polychronakis,[2] and Sotiris Ioannidis [1]

[1] Institute of Computer Science, Foundation for Research and Technology—Hellas, Greece
[2] Columbia University, USA

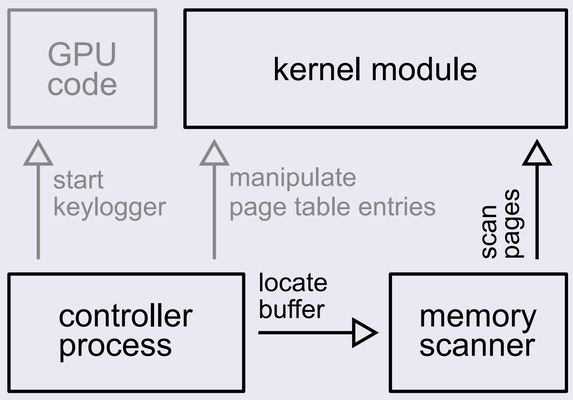
{ladakis,koromil,gvasil,sotiris}@ics.forth.gr, mikepo@cs.columbia.edu

Motivation

- How can we hide malicious code from anti-virus and -malware software?
- Can we leverage the GPU to build stealthier malware?

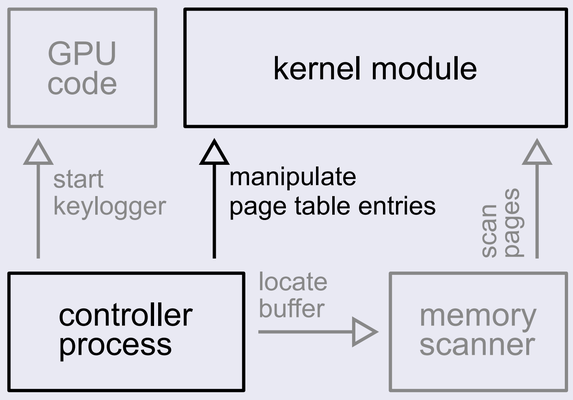
Approach Step by Step

1



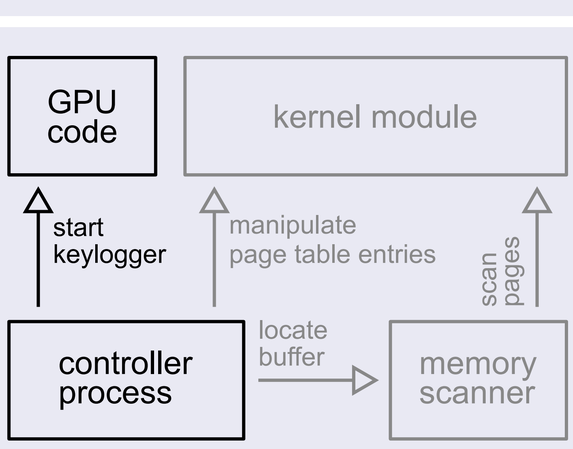
- Scan kernel memory to locate the keyboard buffer
  - Heuristics based on [3]

2



- Map the memory page of the buffer to user space
- Create a device mapping to its physical address using the CUDA API
- Unmap the memory in order to leave no traces
  - All further host accesses result in a segfault

3

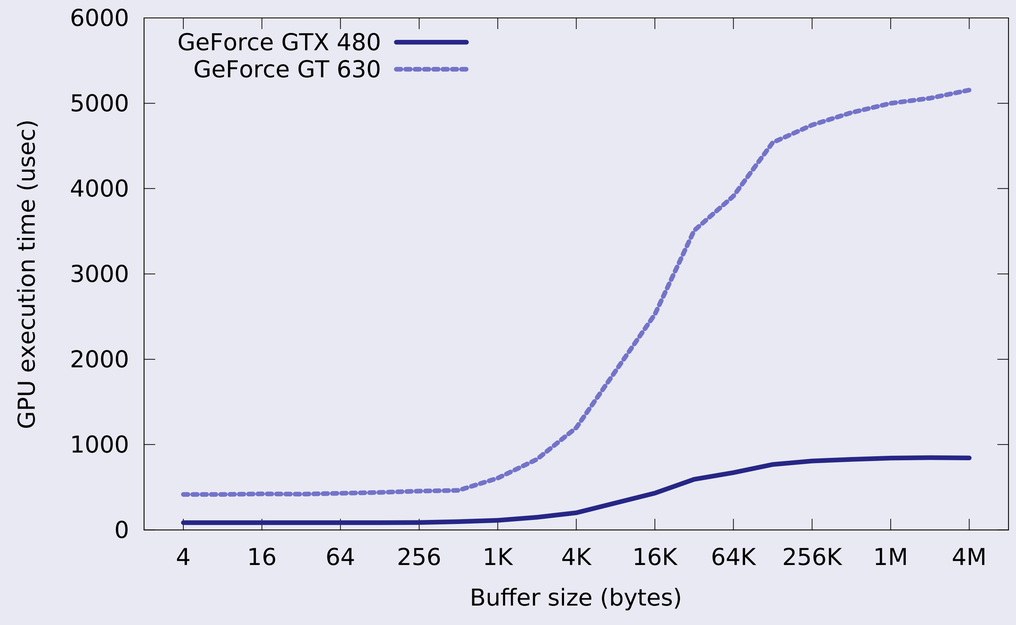


- Periodically spawn a GPU thread that reads the buffer and saves scan codes on device memory as ASCII characters
- Once the log is "big enough" do some computation

Countermeasures

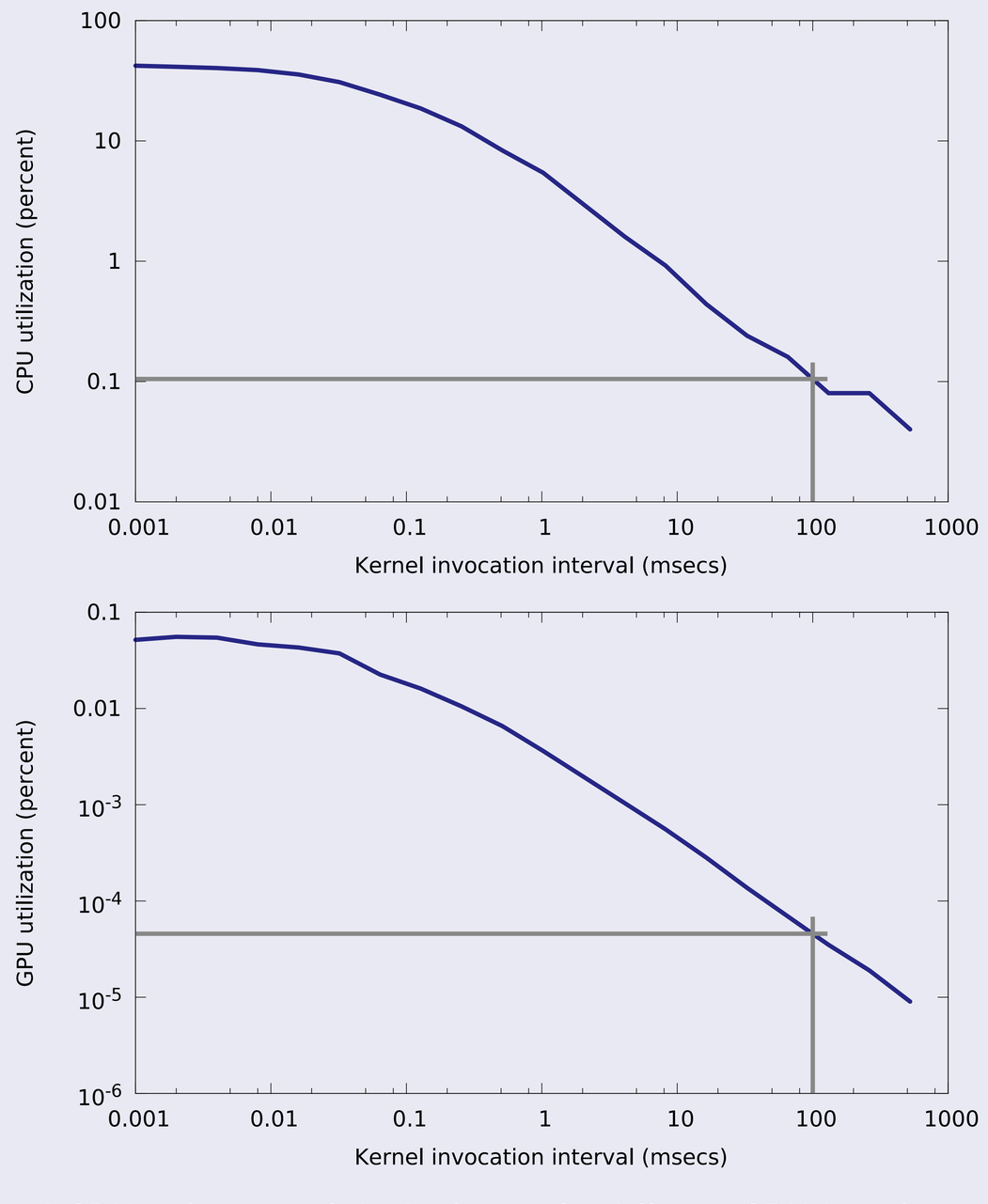
- Manual code analysis
  - Since CUDA version 5.0, `cuda-gdb` can attach to a running process, and inspect the state of the GPU at any point
- Monitor GPU access patterns to detect repeated DMAs between host and device memory
- Profile the GPU utilization

Grep Credit Card Numbers



Execution times for low-end (GT 630) and high-end (GTX 480) graphics cards, when extracting credit card numbers using regular expressions [4] for different captured data sizes.

Runtime Overhead



CPU and GPU utilization of the keylogger for different GPU kernel invocation intervals. Typically, the duration of a single keypress varies from 100 ms for faster typists, to over one second for slower typists [1].

Limitations and Future Work

- ✘ Requires a CPU process to control its execution
- ✔ An attacker can hide the CPU component by injecting its code into the address space of an existing benign process
- Figuring out a way to continue executing on the GPU without the presence of a host context is part of future work

- ✘ Administrative privileges are needed for initializing the environment
- ✔ The kernel module is completely removed afterwards
- ✔ Does not need to hook any code or manipulate any data structures for hiding its presence

References

[1] David Kieras.
Using the Keystroke-Level Model to Estimate Execution Times.
University of Michigan, 2001.

[2] Evangelos Ladakis, Lazaros Koromilas, Giorgos Vasiliadis, Michalis Polychronakis, and Sotiris Ioannidis.
You Can Type, but You Can't Hide: A Stealthy GPU-based Keylogger.
In *Proceedings of the 6th European Workshop on System Security*. EuroSec, Prague, Czech Republic, April 2013.

[3] Patrick Stewin and Iurii Bystrov.
Understanding DMA Malware.
In *Proceedings of the 9th Conference on Detection of Intrusions and Malware & Vulnerability Assessment*. DIMVA, Heraklion, Crete, Greece, July 2012.

[4] Giorgos Vasiliadis, Michalis Polychronakis, and Sotiris Ioannidis.
Parallelization and characterization of pattern matching using GPUs.
In *Proceedings of the 2011 IEEE International Symposium on Workload Characterization*, IISWC, 2011.

This work was presented at EuroSec'13 and was supported in part by the FP7 projects MALCODE and SysSec, funded by the European Commission under Grant Agreements No. 254116 and No. 257007, and by the project ForToo, funded by the Directorate-General for Home Affairs under Grant Agreement No. HOME/2010/ISEC/AG/INT-002.