Regular Expression Matching on Graphics Hardware for Intrusion Detection

Giorgos Vasiliadis, Michalis Polychronakis, Spiros Antonatos,
Sotiris Ioannidis, Evangelos P. Markatos
FORTH-ICS, Greece

RAID'09, 25 September 2009

Overview

- Increase the processing throughput of network intrusion detection systems (NIDS)
- Offload pattern matching operations to the **GPU**
  - previous works: string searching
  - this work: **Regular expression matching**

Outline

- Introduction
- Regexp matching on the GPU
- Performance evaluation
- Summary

Motivation

- Pattern matching accounts for up to **80%** of the total CPU processing time in modern NIDS
- Graphics Cards
  - Easy to program
  - Powerful and ubiquitous
  - Vendors have started promoting GPUs as general-purpose computational units

- *Why not using the spare cycles of the GPU to speed up NIDS operations?*
  - String searching on the GPU [Jacob ’06, Goyal ’08, Vasiliadis ’08]

Regular Expressions

- Much more flexible and expressive compared to string signatures

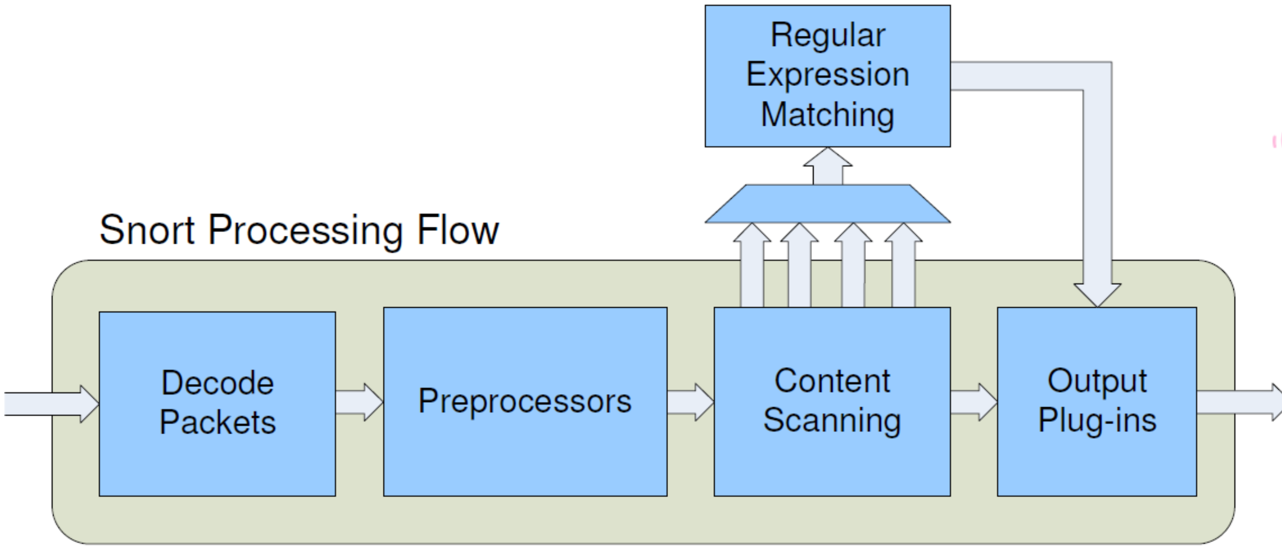
- 45% of the rules in Snort v2.6 use regular expressions

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 10202:10203 (msg:"CA
license GCR overflow attempt"; flow:to_server,established;
content:"GCR NETWORK<"; depth:12; offset:3; nocase;
pcre:"/^\S{65}|\S+\s+\S{65}|\S+\s+\S+\s+\S{65}/Ri"; sid:3520;)
```

- Regular expression matching is much more expensive in terms of CPU cycles than string searching

  Perfect for off-loading to the GPU

Regular Expressions in Snort



- Each expression is compiled into a separate automaton
- Implemented using the PCRE library
- String searching pre-filtering to skip regex matching in the common case

```
alert tcp any any -> any 80 (content:"<OBJECT"; nocase;
pcre:"/<OBJECT\s+[^>]*type\s*=[\x22\x27]\x2f{32}/smi";)
```

Regular Expression Implementations

- **NFA** (Non-deterministic Finite Automata)

  for a given state and input byte, there may be several possible next states

  ✓ Compact representation

  ✓ Greedy or lazy matching, back-references (backtracking)

  ✘ Searching can be exponentially slow (backtracking)

- **DFA** (Deterministic Finite Automata)

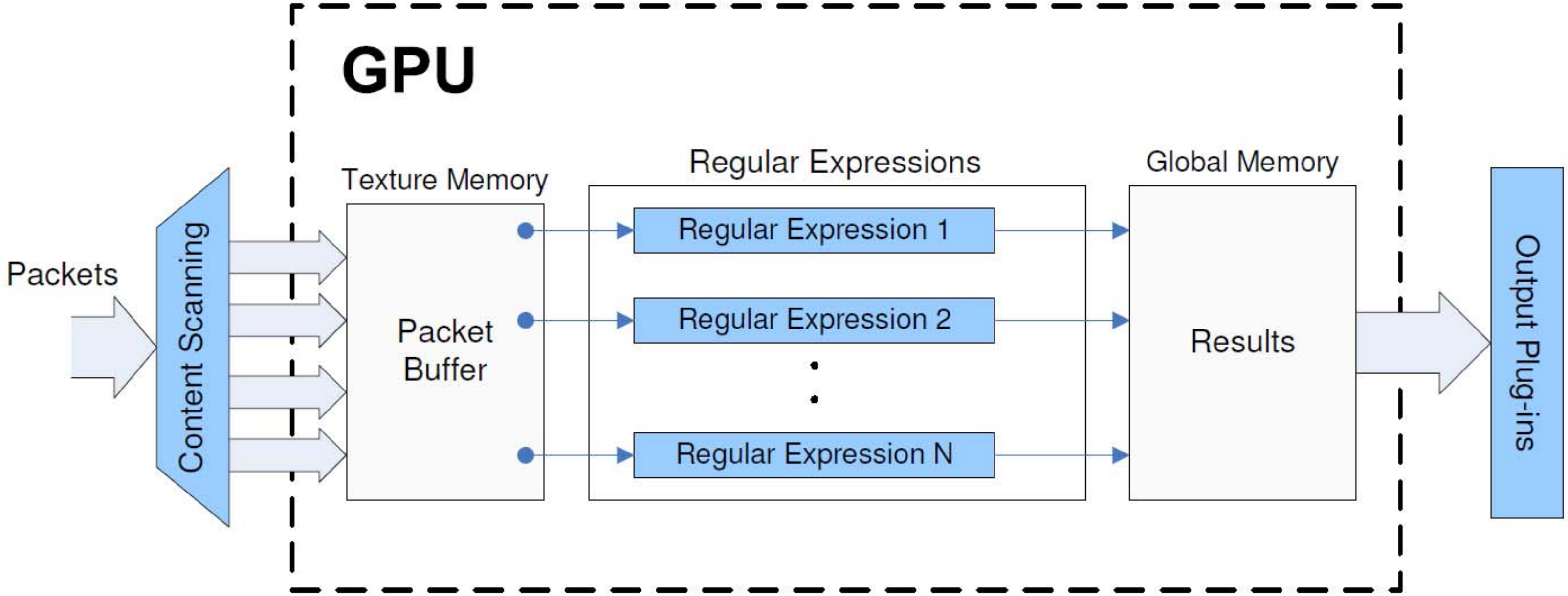
  for a given state and input byte, there is only one next state

  ✘ Can consume an exponentially large amount of memory

  ✘ Greedy matching only (no backtracking)

  ✓ **Searching is fast – O(N)** (no backtracking)

Regular Expression Matching on the GPU



- GPU operates in a SPMD fashion
  - Ideal for creating multiple instances of finite state machines
- Regexps are compiled to DFAs at start-up
  - Run on different stream processors, operate on different data

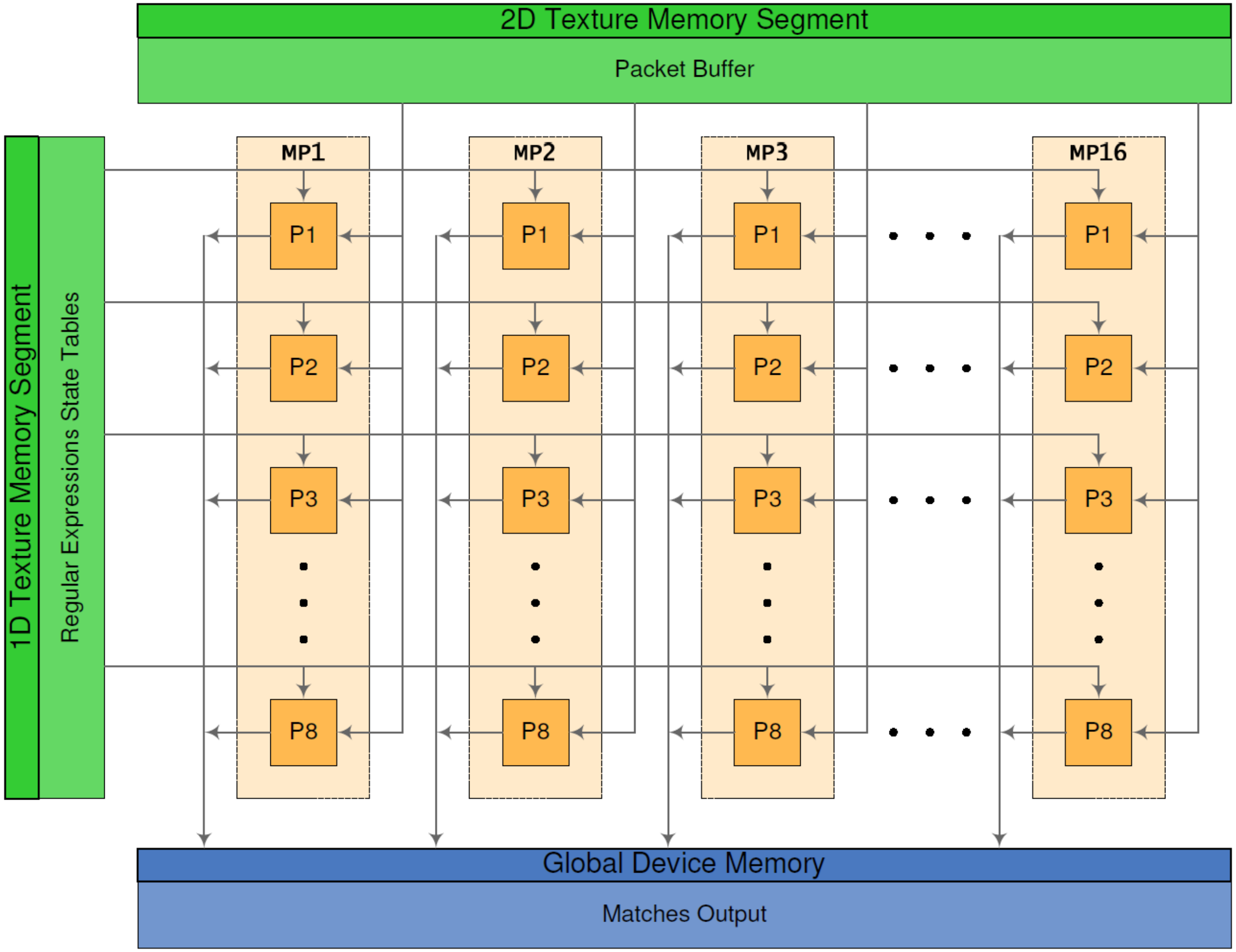
Transferring Packets to the GPU

- Packets are transferred to the GPU in batches

0	4	6 … 1536
Reg.Ex. ID	Length	Payload
Reg.Ex. ID	Length	Payload
Reg.Ex. ID	Length	Payload
⋮	⋮	⋮
Reg.Ex. ID	Length	Payload

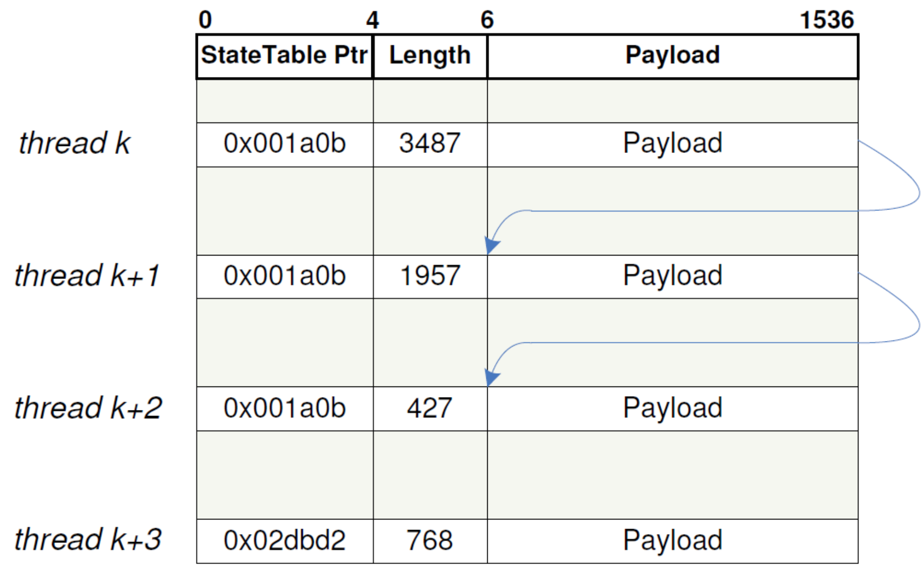
- Copies are performed using DMA, without occupying the CPU
  - *Double-buffering* allows for computation and communication to overlap

GeForce 9800 GX2 with 128 stream processors



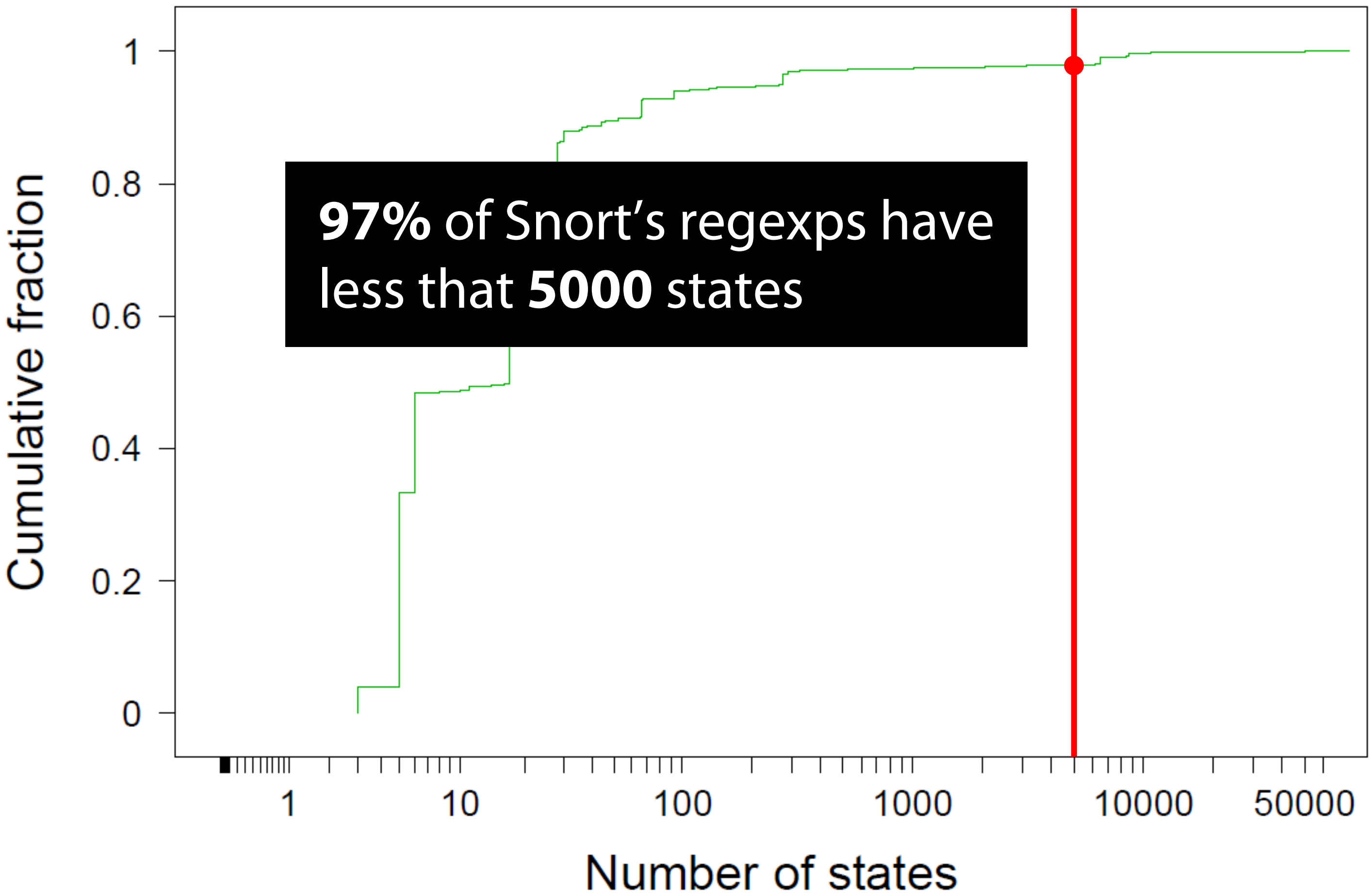
Handling Reassembled TCP streams

- Need to match patterns that span multiple packets
  - 64K pseudo-packets
- Split into MTU-sized packets in consecutive rows in the buffer



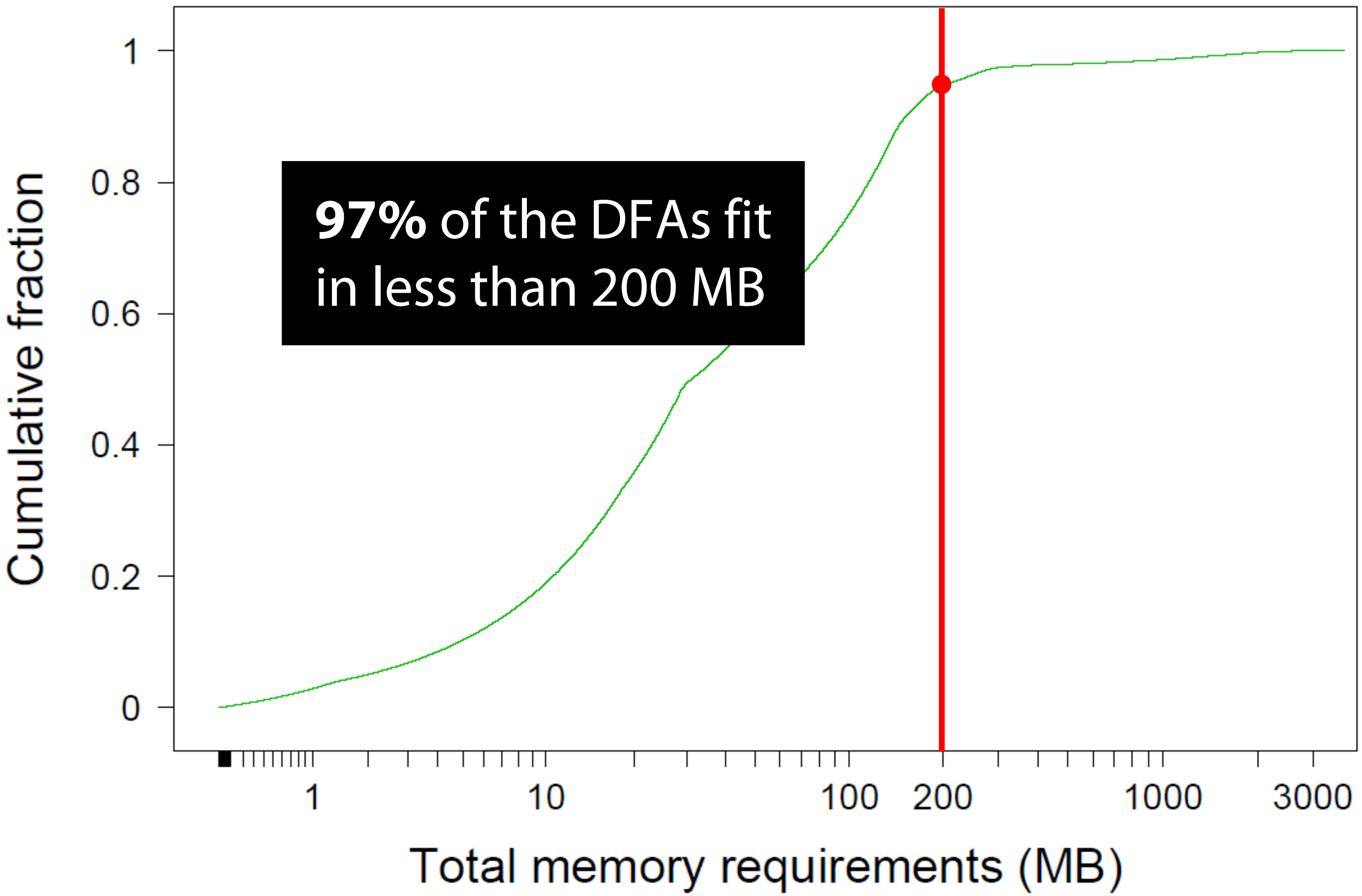
- A thread continues searching in following rows until a final or fail state is reached

DFAs: Number of States



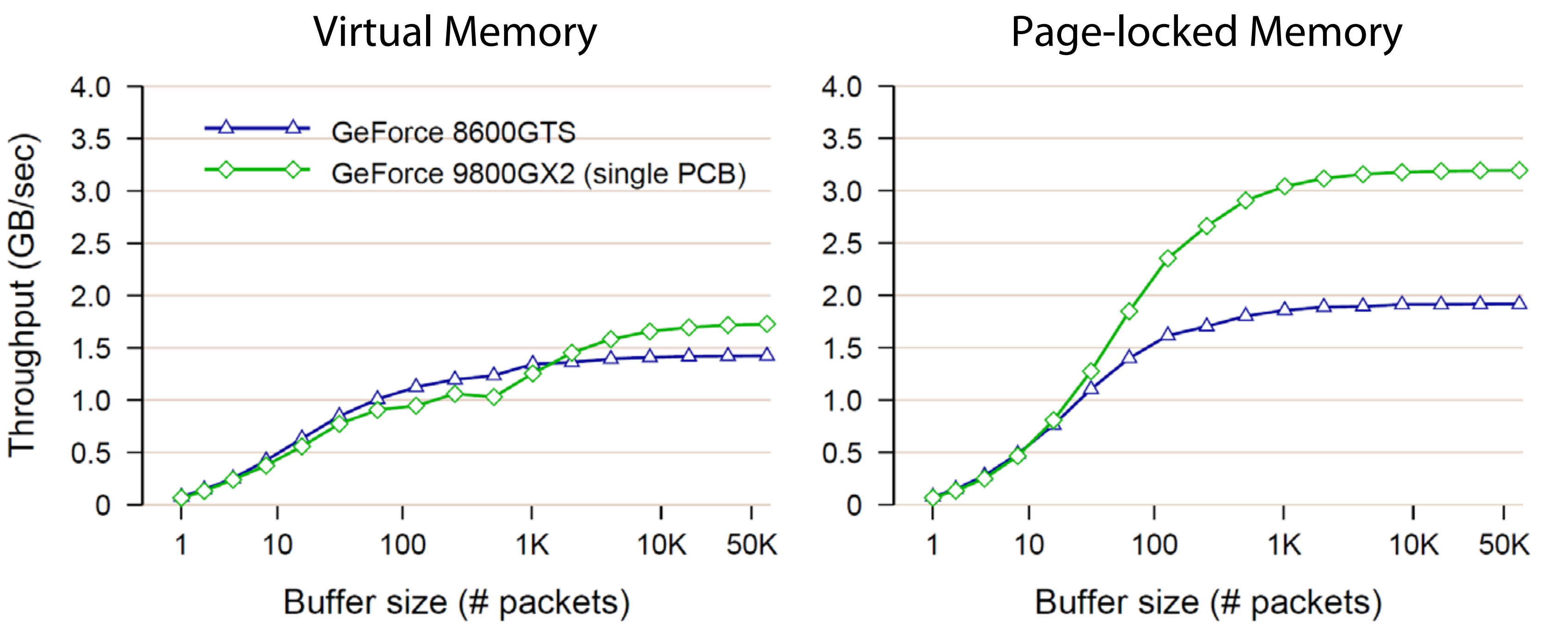
- *11,775 regexps in Snort v2.6*

DFAs: GPU Memory Requirements



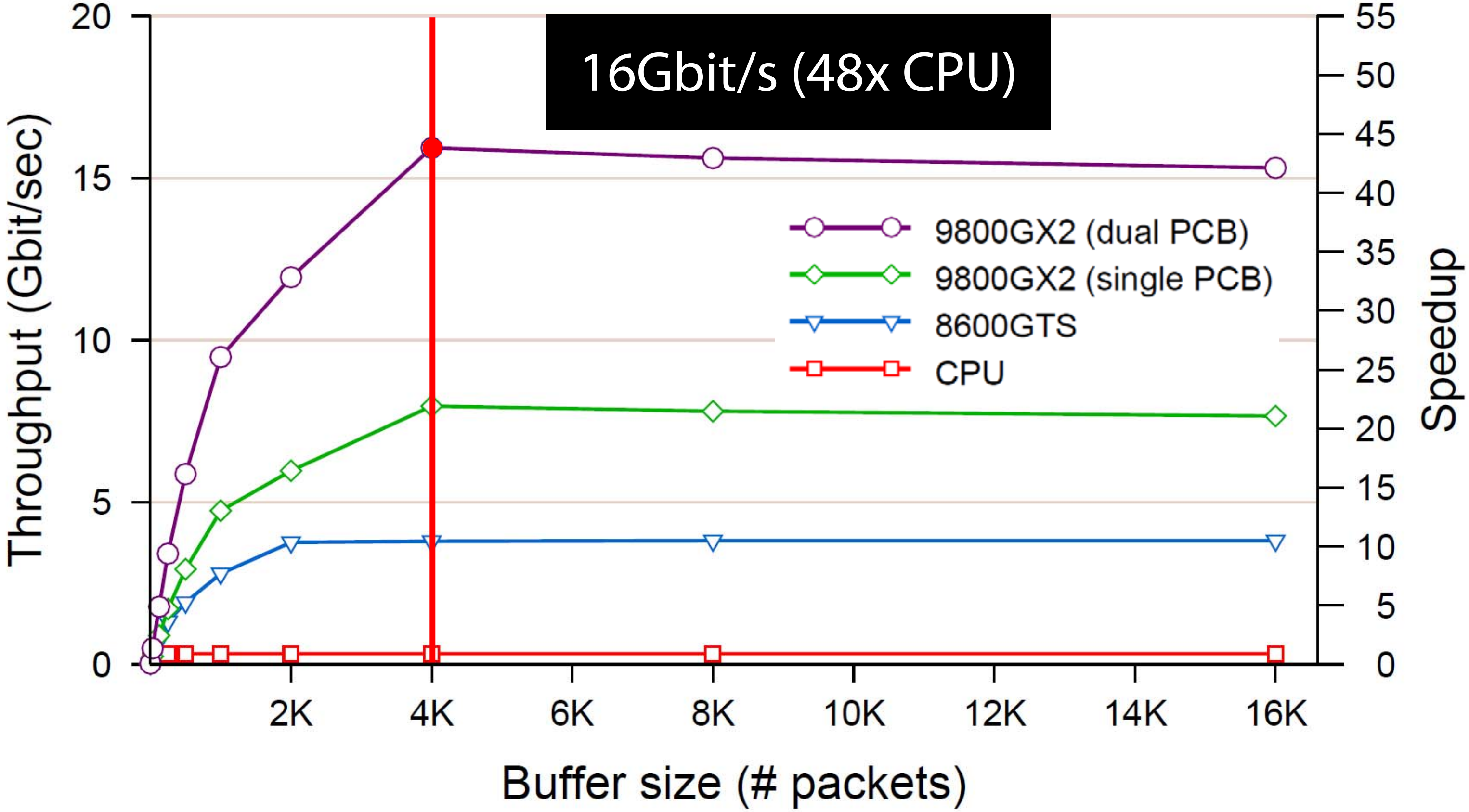
- *The rest 3% is matched on the CPU using NFAs*

CPU → GPU Packet Transfer Throughput



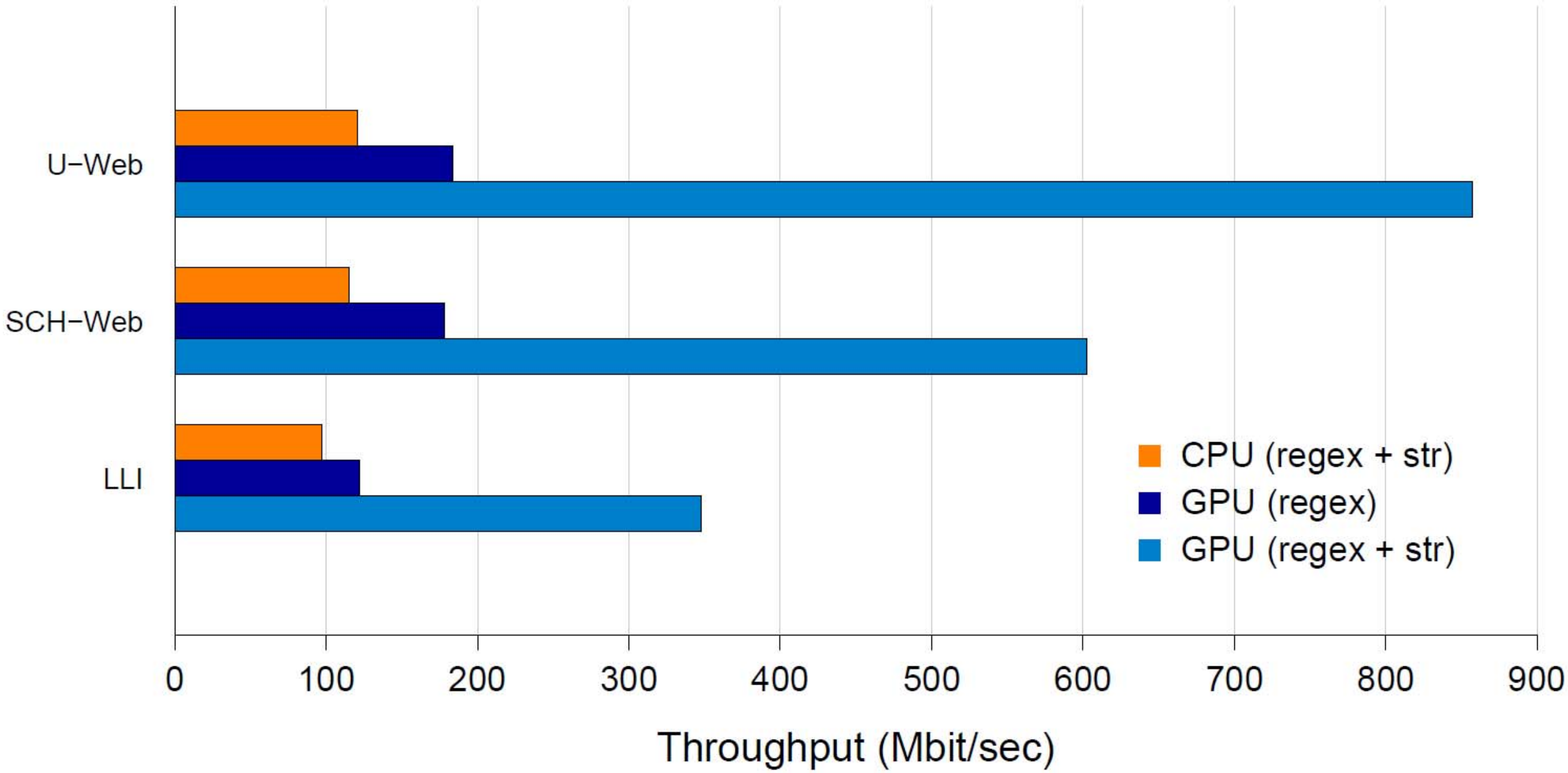
- Use page-locked memory to store incoming packets
- DMA allows for higher transfer throughput

GPU Raw Processing Troughput



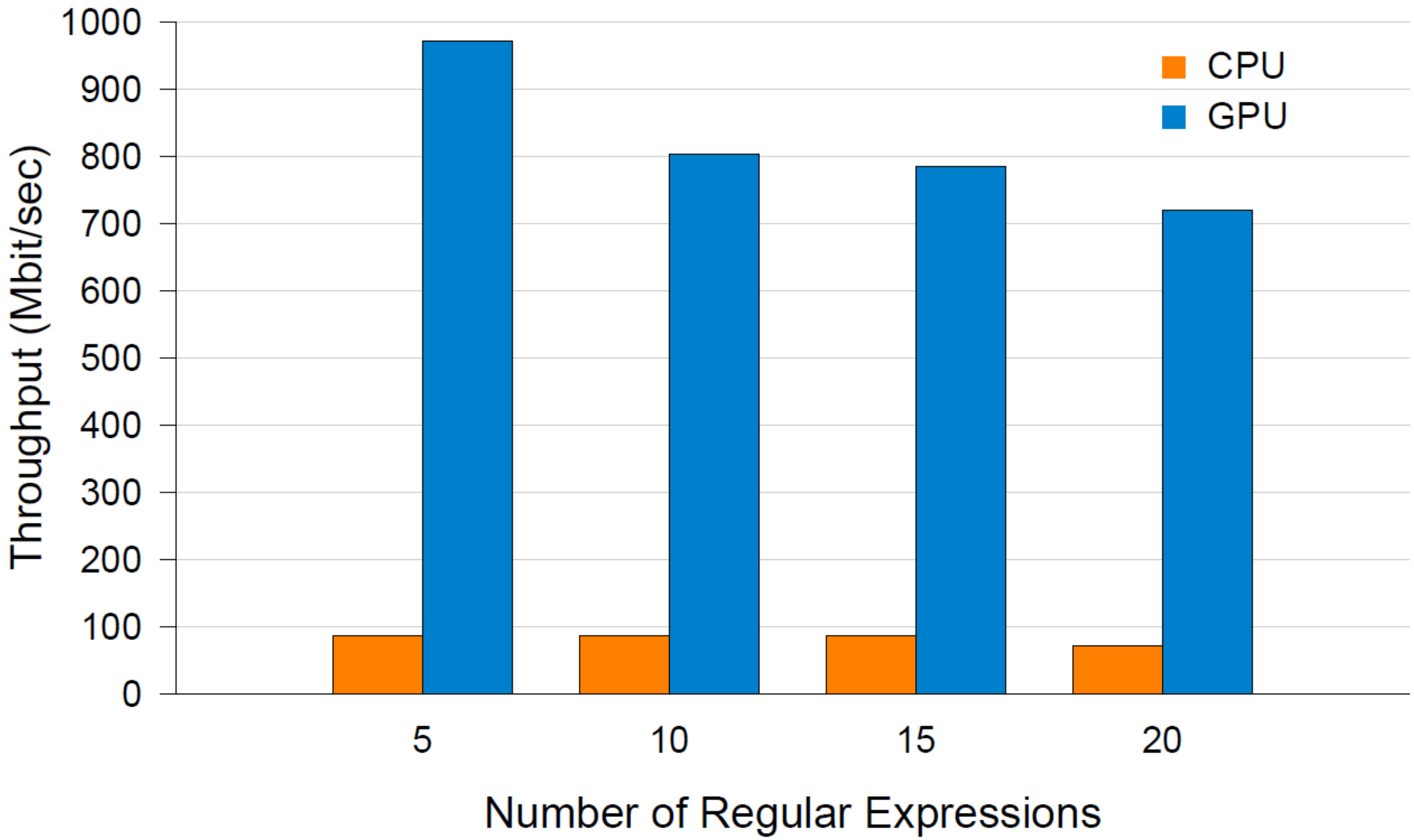
- *Storing the state machines tables into texture memory achieves better performance (due to caching)*
- *The cost of transferring the packets to the GPU space is not included*

Snort Processing Throughput



- *LLI trace performance is reduced due to extensive TCP stream reassembly*
- *The single-threaded design of Snort forces us to use only one PCB (half of the card's computing power)*

Snort Processing Throughput (Pure Regex)



- *Web-traffic only, removed all "content:" operators*
- *Each packet is checked against all regexps*

Summary

- Regex matching on the GPU is practical…
- …and **fast!**
  - 16Gbit/s raw throughput (48x CPU)
  - up to 800Mbit/s (8x CPU) when applied in Snort
- Future work
  - Multiple threads/Snort instances (utilize both PCBs)
  - Alternative implementations (single/few DFAs, xFAs, speculation – next presentation)
  - Multiple graphics cards (lots of space in the box)

Regular Expression Matching on Graphics Hardware for Intrusion Detection

thank you!

Giorgos Vasiliadis, gvasil@csd.uoc.gr
Michalis Polychronakis, mikepo@ics.forth.gr
Spiros Antonatos, antonat@ics.forth.gr
Sotiris Ioannidis, sotiris@ics.forth.gr
Evangelos Markatos, markatos@ics.forth.gr