Implementation of a Content-Scanning Module for an Internet Firewall

James Moscola, John W. Lockwood, Ronald P. Loui, Michael Pachos

April 9, 2003

Research Sponsored by:

Applied Research Laboratory
Department of Computer Science and Engineering

`http://www.arl.wustl.edu/arl/projects/fpx`
`http://www.globalvelocity.info`

Outline

- **Introduction**
- **Motivation**
- **Regular Expression Scanning**
  - NFA vs. DFA
- **Firewall Module**
  - Circuit Implementation
  - Automated Tool Flow
- **Applications**
  - Network Security
  - Virus Protection
  - Copyright Protection
  - SPAM Filter
- **Conclusion**

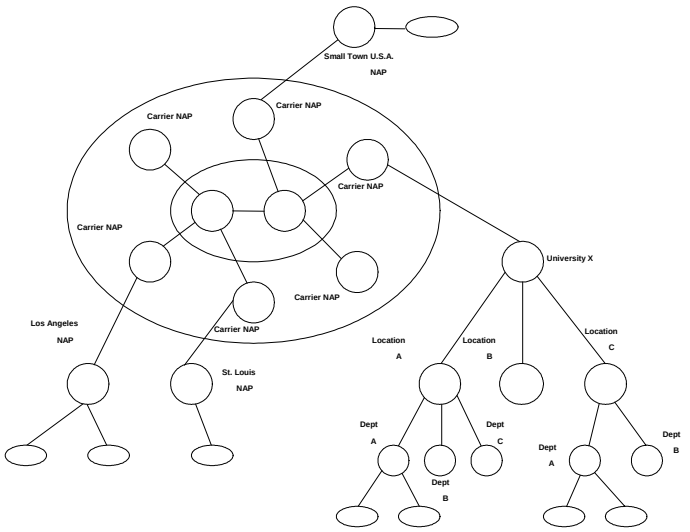
Introduction

- **Internet firewalls and intrusion detection systems have become critical components of the Internet**
- **Most current hardware only process the packet headers**
- **Many viruses, digital media and certain denial of service attacks can only be detected by scanning the payload of a packet**
- **Regular expressions are powerful for searching**

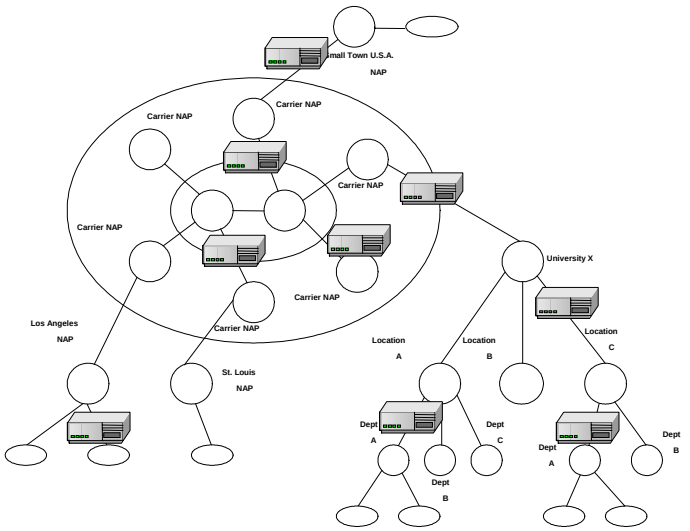
Motivation

- **Problem Description:**
  - Difficult to control flow of data in peer-to-peer networks
    - Movies
    - Music
    - Computer viruses
    - Denial of Service (DoS) attacks
    - SPAM
- **Solution:**
  - Scans packet payloads to detect digital signatures
    - Prevent unlawful distribution of digital content
    - Quarantine and eliminate viruses
    - Prevent DoS attacks and SPAM from spreading
  - Operate at the speed of the network backbone
    - Gigabits/second rates and faster

Controlling Digital Content



Controlling Digital Content



Regular Expression Scanning

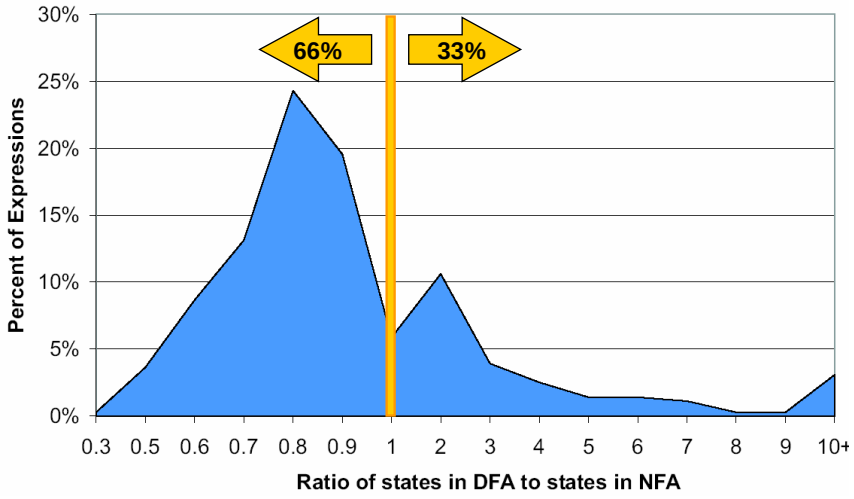
Regular Expressions in FPGAs

- **A couple different approaches …**
  - Nondeterministic Finite Automata (NFAs)
    - Sidhu, Prasanna ; Franklin, Carver, Hutchings
    - Natural parallelism fits nicely into hardware
    - Easy construction
    - Small size
  - Deterministic Finite Automata (DFAs)
    - Theoretically large; but small in practice
    - One active state makes binary encoding possible
    - Compact state representation suitable for network
      - Context of a flow must be loaded / unloaded every packet.

Comparing DFA and NFA sizes

- **Used SPAMAssassin's rule database to compare the sizes of NFAs and DFAs** ***(SPAMAssassin v2.60)***
- **Processed the 358 REs in the database with JLex and compared the outputs**
  - U\.?S\.?(D\.?)?[\ ]*(\$[\ ]*)?([0-9]+,[0-9]+,[0-9]+|[0-9]+\.[0-9]+\.[0-9]+|[0-9]+(\.[0-9]+)?[\ ]*milli?on)
    - NFA = 78 states
    - DFA = 24 states
    - Ratio of DFA:NFA =
      - 24 : 78 =
      - 0.3 : 1

Comparing DFA and NFA sizes

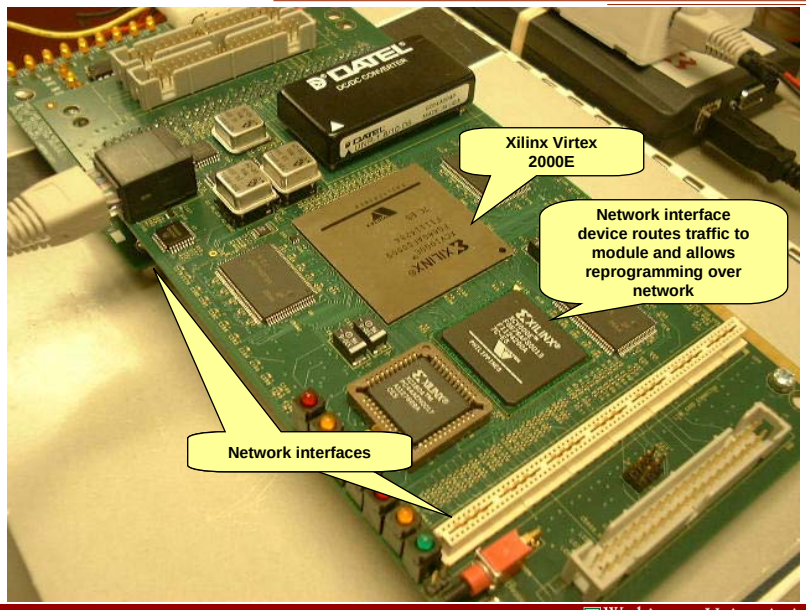


Content Scanner
Circuit Implementation

Requirements for Content Scanner

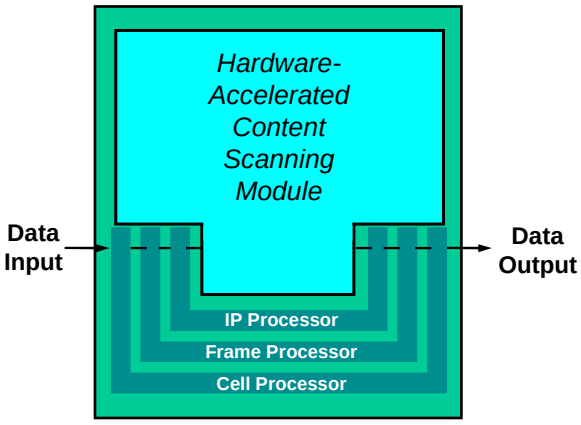
- **Need the ability to…**
  - Scan every character of every packet's payload
    - To find regular expressions
  - Actively drop packets
    - That match a given expression
  - Generate an alert message
    - To Identify which expressions in a given set matched
  - Send an alert message to a log server
    - When a match is detected
  - Easily reconfigure the scanner
    - To search for a new set of expressions

Photograph of the FPX



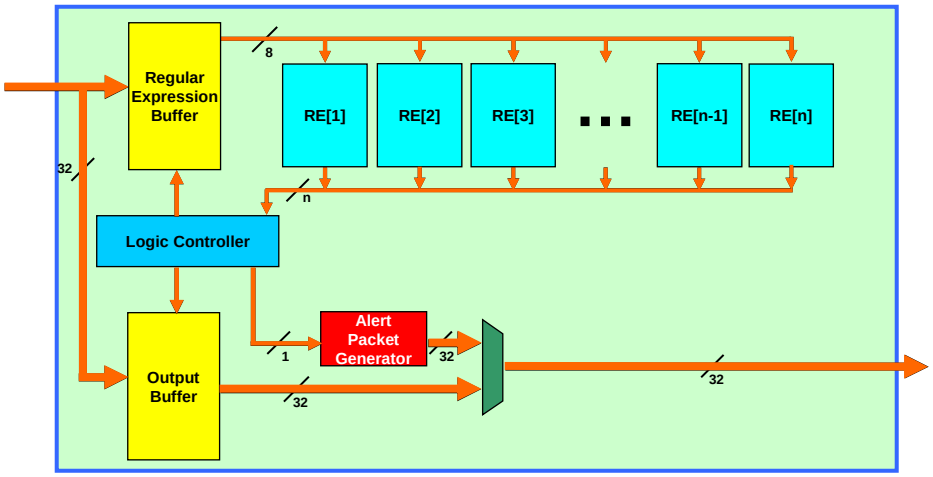
Infrastructure

- **Processing Module**
  - Processes Data passing through the module
- **Protocol Wrappers***
  - Segment and reassemble Internet packets
  - Compute packet headers, lengths, and checksums
- **Interfaces**
  - Read and write packets to network



*FPL '01

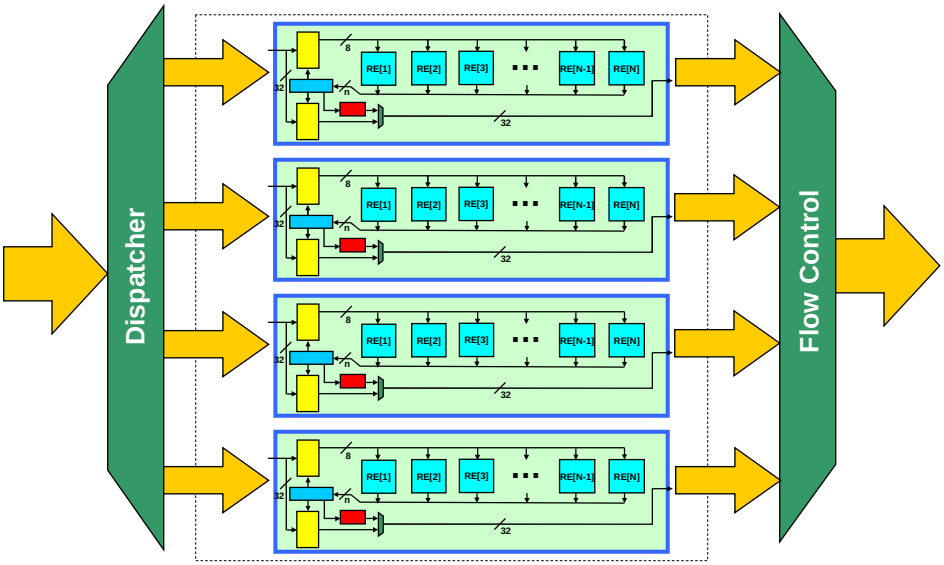
Content Scanner Module



Increasing Throughput via Parallelism

- **Processing packets at the full line rate**
  - Problem:
    - Single scanner only processes 8 bits per clock cycle
    - Input rate is 32 bits per clock cycle
  - Solution:
    - Four parallel content scanners

Increased Throughput via Parallelism

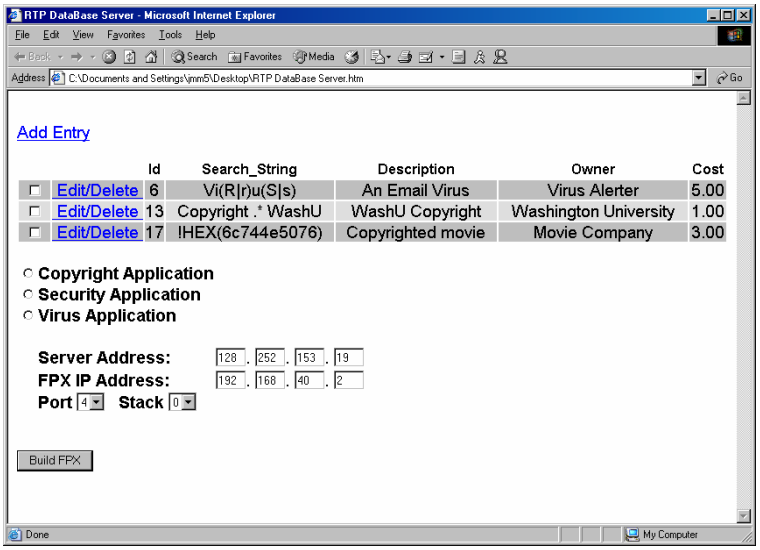


Four Parallel Scanners

Generating the Hardware

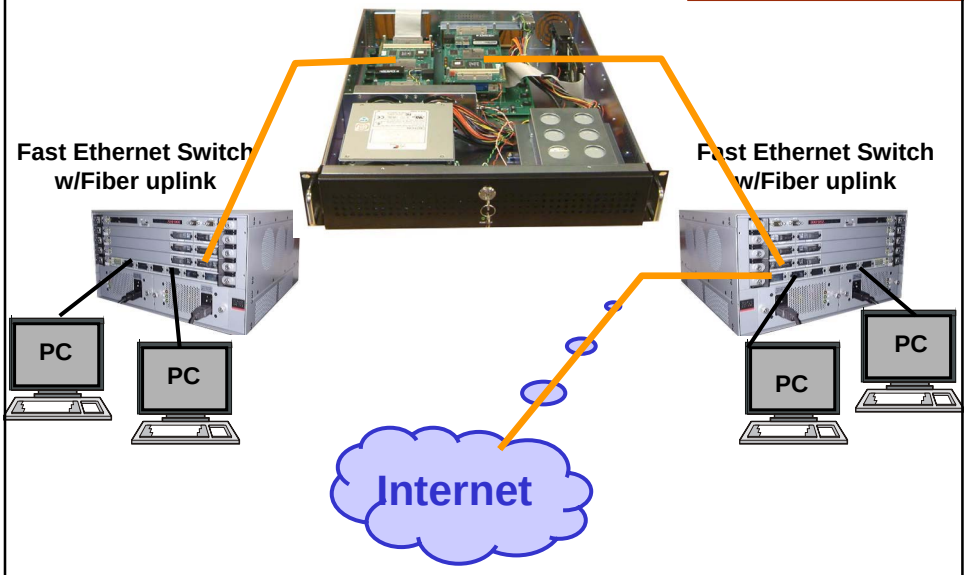
- **Read input specification**
  - Syntax:
    - / expression / property_id(id #) /
  - Example
    - / Vi(R|r)u(S|s) / property_id(6) /
- **Parse with JLex**
  - Create optimized DFA
- **Convert DFA to VHDL**
  - Create structural component to connect all DFAs
- **Synthesize and place and route**
- **Dynamically reprogram the FPGA on the FPX**

Web Configuration Interface

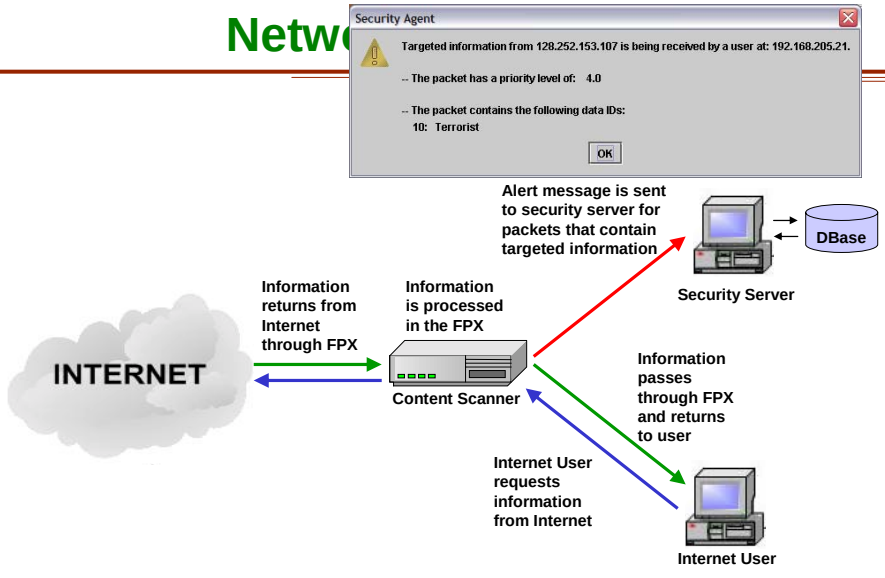


Test Applications

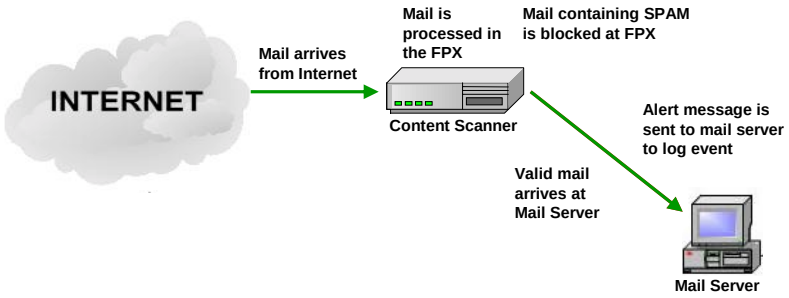
Test Environment



Netw



SPAM Filter



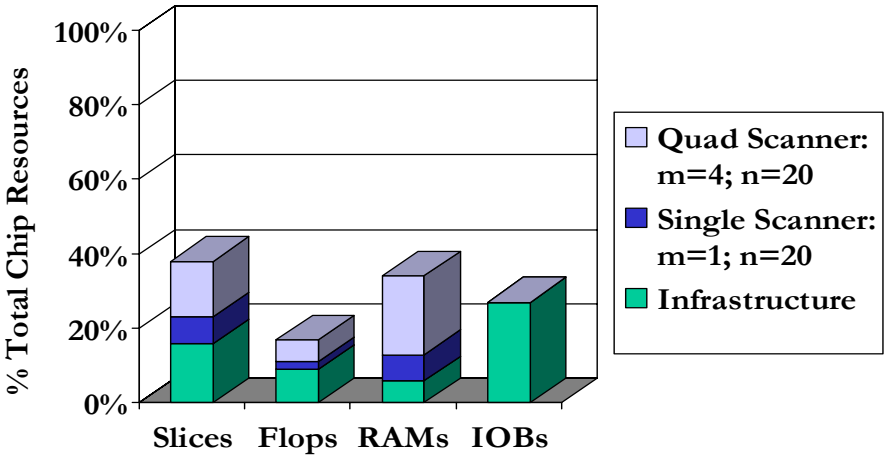
Results

-SPAM Filter

SPAM List

- **General SPAM**
  - "(A|a)(M|m)(A|a)(Z|z)(I|i)(N|n)(G|g)"
  - "CALL NOW"
  - "(L|l)imited (T|t)ime (O|o)ffer"
- **Save Money SPAM**
  - "(C|c)onsolidate"
  - "(F|f)(U|u)(L|l)(L|l) (R|r)(E|e)(F|f)(U|u)(N|n)(D|d)"
- **Fast Money SPAM**
  - "MAKE MONEY FAST"
  - "(W|w)ork from home"
- **Chains and Forwards**
  - "Read this"
  - "FWD"
- **Jokes**
  - "(J|j)oke"
  - "walks into bar"
- **Work List**
  - "(H|h)omework"
  - "(M|m)achine (P|p)roblem"
  - "(C|c)(S|s)536"
  - "Lockwood"
  - "Washington University"
- **Personal List**
  - "(M|m)om"
  - "(D|d)ad"
  - "(C|c)all (H|h)ome"
- **Urgent**
  - "(U|u)(R|r)(G|g)(E|e)(N|n)(T|t)"
  - "Emergency"

Device Utilization – SPAM Filter



Throughput

- **A scanner processes 8-bits per clock cycle**
  - SPAM filter
    - Single scanner
      - 8 bits * 37 MHz = 296 Mbps
    - Quad scanner
      - 8 bits * 37 MHz * 4 scanners = 1.184 Gbps
  - Simple RE circuit
    - Single scanner
      - 8 bits * 80 MHz = 640 Mbps
    - Quad scanner
      - 8 bits * 80 MHz * 4 scanners = 2.5 Gbps

Conclusion

- **Module has been implemented on the FPX**
  - That integrates into a firewall
  - Enables full processing of packet payloads
  - Module is capable of…
    - Passively reviews packets
    - Actively drops packets
    - Generates alert messages to notify of a match
- **Design Flow has been created**
  - Maintains strings in database table
  - Automatically generates bitfiles
- **Module has been tested**
  - Operates with real Internet traffic
  - Module Operates at speeds of 1.2 Gbps - 2.5 Gbps
  - On display for demo night

Current Work

- **Optimize the circuit to achieve better speed**
- **Behavior on regular expression basis instead of chip**
- **Combine with TCP-Splitter* to process data on a stream-by-stream basis instead of on a packet-by-packet basis**
  - requires ability to load/unload stream state of the DFAs

*Hot Interconnects '02

Acknowledgements

- **Washington University**
  - **Applied Research Lab**
    - **Faculty**
      - John Lockwood
      - Jon Turner
    - **Graduate Students**
      - David Taylor
      - Todd Sproull
      - Sarang Dharmapurikar
      - David Lim
      - James Moscola
      - David Schuehler
      - Chris Neely
      - Chris Zuver
      - Haoyu Song
      - Henry Fu (Now at Stanford)
      - Bharath Madhusudan
    - **Undergraduate Students**
      - Harvey Ku (at CMU)
      - Eliot Sinclair
      - Mike Attig
      - Doug Stirrut
      - Tucker Evans (Now at General Dynamics)
      - Mike Wrighton (Now at CalTech)
- **Industry Research Partners**
  - David Parlour (Xilinx)
  - Matthew Kulig (Global Velocity)
- **University Research Partners**
  - Prabhu Kuttiyam (University of Kentucky)
  - Ken Calvert (University of Kentucky)
  - Matt Sanders (Georgia Tech)
  - Ron Srodawa (Oakland University)
  - Haiyan Qiao (NDSU)
  - William Perrizo (NDSU)
  - Kuo-Tung Kuo (University of Maryland)
  - Cary Colwell (Naval Postgraduate School)
  - John Gibson (Naval Postgraduate School)
  - Huaiyu Liu (University of Texas at Austin)
  - Qing Tan (University of Toledo)
  - Sachin Shetty (University of Toledo)
  - Rajanikanth Batchu (Mississippi State)
  - Ravi Sankar (USF)
  - Simon Wong (UCLA)
  - Sven Shepstone (University of Cape Town, South Africa)
- **Visiting Faculty and Students**
  - Edson Horta (Univ. de Sao Paulo, Brasil)
  - Florian Braun (University of Stuttgart)
  - Carlos Macian (University of Stuttgart)

More Information

- http://www.arl.wustl.edu/arl/projects/fpx/
- http://www.globalvelocity.info